

**OPTIMASI KOMPOSISI BAHAN MAKANAN ATLET
OLAHRAGA MENEMBAK DENGAN MENGGUNAKAN
METODE *EVOLUTION STRATEGIES* (ES)**

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Nuraini Anitasari
NIM: 135150200111025



PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

OPTIMASI KOMPOSISI BAHAN MAKANAN ATLET OLAHRAGA MENEMBAK
DENGAN MENGGUNAKAN METODE *EVOLUTION STRATEGIES* (ES)

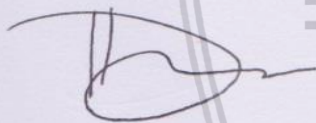
SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Nuraini Anitasari
NIM: 135150200111025

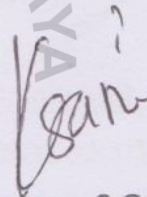
Skripsi ini telah diuji dan dinyatakan lulus pada
2 Agustus 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I



Dian Eka Ratnawati, S.Si., M.Kom
NIP: 19730619 200212 2 001

Dosen Pembimbing II



Titis Sari Kusuma, S.Gz., M.P
NIP: 19800702 200604 2 001

Mengetahui
Ketua Jurusan Teknik Informatika



Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 20 Juli 2018

METERAI
TEMPEL

34960AFF198628507

6000
ENAM RIBU RUPIAH

Nuraini Anitasari

NIM: 135150200111025

DAFTAR RIWAYAT HIDUP

Curriculum Vitae



Data Pribadi

- | | | |
|--------------------------|---|--|
| 1. Nama | : | Nuraini Anitasari |
| 2. Tempat, Tanggal Lahir | : | Jakarta, 04 Agustus 1995 |
| 3. Jenis Kelamin | : | Perempuan |
| 4. Agama | : | Islam |
| 5. Status Pernikahan | : | Belum Menikah |
| 6. Warga Negara | : | Indonesia |
| 7. Alamat | : | Perum. Villa Dieng Residence Blok G No 2, Pisang Candi, Sukun Kota Malang, Jawa Timur 65146 |
| 8. Nomor Telepon / HP | : | 082234722698 |
| 9. e-mail | : | nuraini.anitasari@gmail.com |

Pendidikan Formal

- | | | |
|---------------|---|--|
| • 2001 – 2007 | : | SDN Percontohan Cipinang Cempedak 02 Jakarta |
| • 2007 – 2010 | : | SMPN 36 Jakarta |
| • 2010 – 2013 | : | SMAN 22 Jakarta (IPA) |
| • 2013 – 2018 | : | S1 Informatika Universitas Brawijaya Malang |

Pendidikan Non Formal

- | | | |
|---------------|---|---|
| • 2010 – 2011 | : | Bong Chandra School of Billionaire |
| • 2014 | : | Seminar Nasional 'Creative Animation & Gaming Industry' |

Pengalaman Organisasi

- | | | |
|------------------------------|---|--|
| • 2007 – 2008 | : | Anggota Divisi Keputrian, OSIS SMPN 36 Jakarta |
| • 2014 – 2015 Universitas | : | Staff Kesekretariatan, UKM Menembak BASIC SHOOTING CLUB Brawijaya Malang |
| • 2014 Kerja | : | Anggota Divisi Humas, Kepanitiaan Ramadhan Corner (Program Kementrian Pengabdian Masyarakat BEM TIIK) Universitas Brawijaya Malang |
| • 2015 – 2016 CLUB | : | Staff Divisi Tembak Sasaran, UKM Menembak BASIC SHOOTING Universitas Brawijaya Malang |
| • 2015 | : | Volunteer Kegiatan PRO ACTIVE SOCIAL ACTION (Program Kerja Kementrian Sosial Masyarakat BEM TIIK) Universitas Brawijaya Malang |
| • 2016 – 2017 CLUB | : | Ketua Divisi Tembak Sasaran, UKM Menembak BASIC SHOOTING Universitas Brawijaya Malang |

Keahlian

- Keahlian komputer seperti Microsoft Office, Windows.
- Bahasa pemrograman JAVA Netbeans dan PHP
- Pengolahan database MySQL
- Olahraga menembak dengan senapan dan pistol

Prestasi

- 2014 : Juara 1 Menembak Bench Rest Putri – Olimpiade Brawijaya
- 2014 : Juara 3 Menembak Metal Silhouette Putri – Olimpiade Brawijaya
- 2015 : Juara 2 Air Rifle Hunting Putri 10 Meter – Brawijaya Shooting Tournament

Demikian Curriculum Vitae ini saya buat dengan sebenar-benarnya, untuk digunakan sebagaimana mestinya.

Malang, 28 Agustus 2018

Hormat Saya,



(NURAINI ANITASARI)



ABSTRAK

Sumber energi utama pada manusia untuk dapat melakukan tumbuh kembang agar lebih optimal didapat dari makanan yang dikonsumsi. Berdasarkan perbedaan kandungan tiap jenis makanan yang terdiri dari karbohidrat, lemak, protein dan sebagainya, maka dibutuhkan diet yang optimal untuk setiap individu. Peran diet optimal ini sangat penting khususnya bagi atlet agar bisa berprestasi. Olahraga menembak merupakan salah satu olahraga yang membutuhkan aspek *endurance* untuk atletnya. Agar dapat berprestasi, selain dengan latihan rutin maka para atlet juga harus mampu mengatur porsi makanan yang dikonsumsi secara optimal. Penelitian ini mengimplementasikan algoritma *Evolution Strategies* untuk mengoptimasi komposisi bahan makanan atlet menembak dengan total bahan makanan sebanyak 125 bahan. Proses reproduksi menggunakan metode *mutation* dan proses seleksi menggunakan *elitism selection*. Berdasarkan hasil dari pengujian parameter, didapatkan ukuran populasi terbaik sebanyak 30 individu, ukuran *offspring* terbaik sebanyak 7 μ , jumlah generasi terbaik sebesar 60 generasi dan dengan rata – rata *fitness* terbaik 0,004904. Sedangkan hasil dari uji coba terhadap 4 studi kasus, diketahui bahwa sistem mampu menghasilkan rekomendasi berat makanan dengan harga minimal dan diketahui pula rata – rata hasil karbohidrat 90,1%, lemak 98,7% dan protein 105,4% dimana asupan karbohidrat serta lemak dikategorikan cukup dan protein dikategorikan baik menurut indikator ahli gizi dalam melakukan penilaian konsumsi pangan.

Kata kunci: *Algoritma Evolution Strategies, Kombinasi Berat Bahan Makanan, Atlet, Olahraga Menembak*

ABSTRACT

The main energy source in humans to be able to grow and develop to be more optimal obtained from the food consumed. Based on differences in each food composition that has been consumed consisting of carbohydrate content, fat content, protein content, energy and so forth then it is needed an optimal diet for each individual. The role of an optimal diet is very important, especially for athletes in order to perform. Shooting sports is one sports that requires endurance aspect for athletes. In addition to regular exercise, so athletes should also be able to manage the portion and food consumption in order to success. In this research, Evolution Strategies algorithm is implemented to optimize the composition of food for shooting athletes with total food ingredients that are used 125 foods. Then reproduction process in this research using mutation method and selection process using elitism selection. Based on the results of the algorithm parameter test, the best population size was obtained by 30 individuals, the size of best offspring as 7 μ , the best generation number of 60 generations and with the best fitness average 0,004904. Meanwhile, based on the trial of case studies conducted 4 times, it is known that the system is able to produce the result of food composition with minimal prices and it is also known that according to the indicator of nutritionist in assessing food consumption, carbohydrate and fat is categorized as sufficient and protein is categorized as good with an average carbohydrate is 90,1%, fat is 98,7% and protein is 105,4%.

Keywords: Evolution Strategies algorithms , Combination of Weight Food , Athletes, Shooting Sports

KATA PENGANTAR

Dengan menyebut nama Allah SWT Yang Maha Pengasih dan Maha Penyayang. Puji syukur kehadiran Allah SWT karena limpahan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan penelitian ini.

Pada kesempatan ini penulis juga ingin menyampaikan terima kasih kepada:

1. Dian Eka Ratnawati, S.Si., M.Kom dan Titis Sari Kusuma, S.Gz., M.P, selaku Dosen Pembimbing yang telah membimbing, mengarahkan dan memberikan banyak saran kepada penulis.
2. Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Universitas Brawijaya.
3. Seluruh Dosen FILKOM UB yang telah membagikan ilmunya dan seluruh Civitas Akademika FILKOM UB yang telah membantu penulis demi kelancaran skripsi ini.
4. Keluarga besar penulis di Jakarta khususnya Chairunnisa, Muhammad Aldy Saputra, Tien Nuryatien, Edi Suwandi, Arista Muhartanto dan A.Y Putrawan B. Serta keluarga besar penulis di Jombang khususnya Sri Susilo Ningsih dan Zainus Sulton atas segala do'a, nasihat, dukungan baik moril maupun materil yang begitu besar terhadap kelancaran dalam menyelesaikan skripsi ini.
5. Alex Roy Kurniawan selaku *partner*, pelatih dan pendiri UKM Menembak BASIC Shooting Club yang telah bersedia meluangkan waktunya dan memberikan dukungan serta referensi/sumber data penelitian.
6. Sahabat serta teman-teman penulis antara lain Fitri, Oka, Fahri, Reno, Syndu, Syawal, Prima, Pipit, Vivin, Adelina, Agung, Assel, Elan, Nadhira, Ivan dan teman – teman lainnya atas bantuan, dukungan, motivasi serta bersedia berbagi informasi demi kelancaran skripsi.
7. Keluarga besar UKM Menembak BASIC Shooting Club Universitas Brawijaya Malang dan PERBAKIN atas kebersamaan, dukungan serta ilmu dan pengetahuan yang telah diberikan.

Dengan kerendahan hati, penulis menyadari bahwa skripsi ini masih memiliki banyak kekurangan. Oleh karena itu kritik dan saran sangat dibutuhkan untuk menyempurnakan skripsi ini agar lebih baik.

Malang, 20 Juli 2018

Penulis

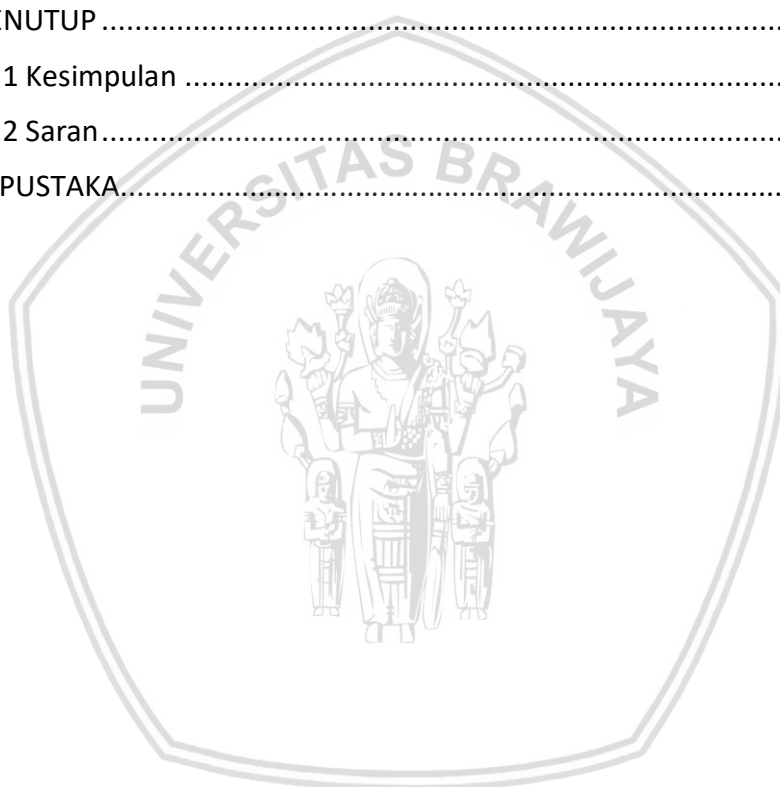
nuraini.anitasari@gmail.com

DAFTAR ISI

| | |
|--|------|
| PENGESAHAN | ii |
| PERNYATAAN ORISINALITAS..... | iii |
| KATA PENGANTAR | iv |
| ABSTRAK | v |
| ABSTRACT | vi |
| DAFTAR ISI | vii |
| DAFTAR TABEL | x |
| DAFTAR GAMBAR | xi |
| DAFTAR PERSAMAAN..... | xii |
| DAFTAR LAMPIRAN | xiii |
| BAB 1 PENDAHULUAN..... | 1 |
| 1.1 Latar belakang..... | 1 |
| 1.2 Rumusan masalah | 3 |
| 1.3 Tujuan | 3 |
| 1.4 Manfaat | 3 |
| 1.5 Batasan masalah | 3 |
| 1.6 Sistematika pembahasan..... | 4 |
| BAB 2 LANDASAN KEPUSTAKAAN | 6 |
| 2.1 Kajian Pustaka | 6 |
| 2.2 Dasar Teori..... | 8 |
| 2.2.1 Olahraga Menembak | 8 |
| 2.2.2 Kebutuhan Gizi Atlet Olahraga Menembak | 9 |
| 2.2.2 Algoritma <i>Evolution Strategies</i> (ES) | 13 |
| BAB 3 METODOLOGI | 19 |
| 3.1 Studi Literatur | 19 |
| 3.2 Pengumpulan Data..... | 20 |
| 3.3 Analisis Kebutuhan Sistem..... | 20 |
| 3.4 Perancangan Sistem | 20 |
| 3.5 Pengujian dan Analisis Sistem | 20 |

| | |
|--|----|
| 3.6 Kesimpulan | 21 |
| BAB 4 PERANCANGAN SISTEM | 22 |
| 4.1 Formulasi Permasalahan | 22 |
| 4.2 Siklus Penyelesaian Masalah | 23 |
| 4.2.1 Penyusunan Kromosom Optimal | 24 |
| 4.2.2 <i>Generate</i> Populasi Awal | 25 |
| 4.2.3 Reproduksi | 26 |
| 4.2.4 Nilai <i>fitness</i> | 27 |
| 4.2.5 Penalti Gizi | 28 |
| 4.2.6 Penalti Harga | 29 |
| 4.2.7 Seleksi | 30 |
| 4.3 Perhitungan Manual | 31 |
| 4.3.1 Parameter Algoritma <i>Evolution Strategies</i> | 32 |
| 4.3.2 <i>Generate</i> Populasi Awal | 32 |
| 4.3.3 Reproduksi | 33 |
| 4.3.4 Perhitungan Penalti Gizi | 35 |
| 4.3.5 Perhitungan Penalti Harga | 37 |
| 4.3.6 Menghitung Nilai <i>fitness</i> | 38 |
| 4.3.7 Seleksi | 38 |
| 4.4 Perancangan Uji Coba dan Evaluasi | 40 |
| 4.4.1 Rancangan Pengujian Ukuran Populasi (μ) dan <i>offspring</i> (λ) | 40 |
| 4.4.2 Rancangan Pengujian Banyak Generasi | 41 |
| BAB 5 IMPLEMENTASI | 42 |
| 5.1 Lingkungan Implementasi | 42 |
| 5.1.1 Lingkungan Perangkat Keras | 42 |
| 5.1.2 Lingkungan Perangkat Lunak | 42 |
| 5.2 Implementasi Sistem | 42 |
| 5.2.1 Pengambilan Data Makanan | 42 |
| 5.2.2 Implementasi <i>Generate</i> Populasi Awal | 43 |
| 5.2.3 Implementasi Proses Mutasi | 44 |
| 5.2.4 Implementasi Menghitung Nilai <i>fitness</i> | 48 |
| 5.2.5 Implementasi Proses Seleksi | 50 |

| | |
|--|----|
| BAB 6 ANALISA DAN PENGUJIAN..... | 53 |
| 6.1 Hasil dan Analisa Pengujian Ukuran Populasi (μ) dan <i>offspring</i> (λ) | 53 |
| 6.2 Hasil dan Analisa Pengujian Banyak Generasi | 56 |
| 6.3 Analisa Hasil | 58 |
| 6.3.1 Studi Kasus 1..... | 58 |
| 6.3.2 Studi Kasus 2..... | 59 |
| 6.3.3 Studi Kasus 3..... | 60 |
| 6.3.4 Studi Kasus 4..... | 61 |
| BAB 7 PENUTUP | 64 |
| 7.1 Kesimpulan | 64 |
| 7.2 Saran..... | 65 |
| DAFTAR PUSTAKA..... | 66 |



DAFTAR TABEL

| | |
|--|----|
| Tabel 2.1 Ringkasan Perbedaan Penelitian Sebelumnya | 7 |
| Tabel 2.2 Rumus <i>Basal Metabolic Rate</i> (BMR) | 10 |
| Tabel 2.3 Kategori Tingkatan Aktivitas Fisik | 11 |
| Tabel 2.4 Contoh <i>generate</i> Populasi Awal | 14 |
| Tabel 2.5 Contoh Rekombinasi | 15 |
| Tabel 2.6 Contoh Mutasi Berdasarkan Rekombinasi Sebelumnya | 16 |
| Tabel 4.1 <i>Generate</i> populasi awal berdasarkan inputan user | 33 |
| Tabel 4.2 <i>Random</i> kromosom dari populasi awal | 33 |
| Tabel 4.3 Individu hasil mutasi dari <i>parent</i> | 34 |
| Tabel 4.4 Penalti gizi pada gen individu pertama | 35 |
| Tabel 4.5 Nilai penalti menu berdasarkan waktu makan | 36 |
| Tabel 4.6 Nilai penalti gizi masing-masing individu | 37 |
| Tabel 4.7 Hasil perhitungan penalti harga | 38 |
| Tabel 4.8 Nilai <i>fitness</i> dari tiap kromosom | 38 |
| Tabel 4.9 Gabungan individu yang masuk ke tahap seleksi | 39 |
| Tabel 4.10 Urutan individu berdasarkan dengan <i>fitness</i> yang terbesar | 39 |
| Tabel 4.11 Individu terpilih untuk dimasukkan ke generasi selanjutnya | 40 |
| Tabel 4.12 Tabel rancangan uji coba banyaknya populasi | 40 |
| Tabel 4.13 Tabel rancangan uji coba ukuran <i>Offspring</i> | 41 |
| Tabel 4.14 Tabel rancangan uji coba ukuran generasi | 41 |
| Tabel 6.1 Pengujian Ukuran Populasi | 53 |
| Tabel 6.2 Pengujian Ukuran <i>offspring</i> | 55 |
| Tabel 6.3 Pengujian Banyak Generasi | 57 |
| Tabel 6.4 Hasil Pemenuhan Gizi Studi Kasus 1 | 59 |
| Tabel 6.5 Hasil Pemenuhan Gizi Studi Kasus 2 | 60 |
| Tabel 6.6 Hasil Pemenuhan Gizi Studi Kasus 3 | 61 |
| Tabel 6.7 Hasil Pemenuhan Gizi Studi Kasus 4 | 62 |
| Tabel 6.8 Hasil Persentase Kecukupan Gizi Berdasarkan Studi Kasus | 62 |
| Tabel 6.9 Hasil Perbandingan Harga Berdasarkan Studi Kasus | 62 |

DAFTAR GAMBAR

| | |
|---|----|
| Gambar 2.1 <i>Pseudocode</i> Algoritma <i>Evolution Strategies</i> | 13 |
| Gambar 3.1 Diagram Alir Optimasi Komposisi Bahan Makanan Atlet Olahraga Menembak dengan Menggunakan Metode <i>Evolution Strategies</i> (ES) | 18 |
| Gambar 4.1 <i>Flowchart</i> Optimasi Gizi Atlet Menembak | 22 |
| Gambar 4.2 <i>Flowchart</i> Proses Penyusunan Kromosom Optimal | 23 |
| Gambar 4.3 <i>Flowchart</i> Proses <i>Generate</i> Populasi Awal | 24 |
| Gambar 4.4 <i>Flowchart</i> Pada Proses Reproduksi | 25 |
| Gambar 4.5 <i>Flowchart</i> Proses Menghitung Nilai <i>fitness</i> | 26 |
| Gambar 4.6 <i>Flowchart</i> Proses Hitung Nilai Penalti Gizi | 42 |
| Gambar 4.7 <i>Flowchart</i> Proses Melakukan Perhitungan Nilai Penalti Harga..... | 43 |
| Gambar 4.8 <i>Fowchart</i> Melakukan Proses Seleksi Populasi Baru | 29 |
| Gambar 5.1 <i>Source Code</i> Pengambilan Data Makanan | 42 |
| Gambar 5.2 <i>Source Code</i> <i>Generate</i> Populasi Awal | 43 |
| Gambar 5.3 <i>Source Code</i> Proses Mutasi | 46 |
| Gambar 5.4 <i>Source Code</i> Menghitung Kandungan Gizi Sebenarnya | 47 |
| Gambar 5.5 <i>Source Code</i> Menghitung Penalti Gizi | 48 |
| Gambar 5.6 Implementasi Menghitung Penalti Harga dan Nilai <i>fitness</i> | 49 |
| Gambar 5.7 Implementasi Proses Seleksi | 50 |
| Gambar 6.1 Grafik Pengujian Ukuran Populasi | 53 |
| Gambar 6.2 Grafik Pengujian Ukuran <i>offspring</i> | 54 |
| Gambar 6.3 Grafik Pengujian Banyak Generasi..... | 57 |

DAFTAR PERSAMAAN

| | |
|-----------------------|----|
| Persamaan (2.1)..... | 10 |
| Persamaan (2.2)..... | 10 |
| Persamaan (2.3)..... | 10 |
| Persamaan (2.4)..... | 10 |
| Persamaan (2.5)..... | 10 |
| Persamaan (2.6)..... | 10 |
| Persamaan (2.7)..... | 10 |
| Persamaan (2.8)..... | 14 |
| Persamaan (2.9)..... | 14 |
| Persamaan (2.10)..... | 14 |
| Persamaan (2.11)..... | 14 |
| Persamaan (2.12)..... | 15 |
| Persamaan (2.13)..... | 16 |
| Persamaan (2.14)..... | 16 |
| Persamaan (2.15)..... | 16 |
| Persamaan (2.16)..... | 16 |
| Persamaan (2.17)..... | 16 |
| Persamaan (2.18)..... | 16 |
| Persamaan (2.19)..... | 16 |
| Persamaan (2.20)..... | 16 |
| Persamaan (2.21)..... | 17 |

DAFTAR LAMPIRAN

| | |
|--------------------------------------|----|
| LAMPIRAN A DAFTAR BAHAN MAKANAN..... | 67 |
| LAMPIRAN B DATA RESPONDEN | 73 |
| LAMPIRAN C MANUALISASI..... | 75 |



PENDAHULUAN

Di dalam bab ini akan berisi mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian serta sistematika penulisan dari penelitian ini.

1.1 Latar belakang

Makanan merupakan sumber energi utama pada manusia serta suatu kebutuhan akan gizi terhadap tiap individu untuk melakukan tumbuh kembang tubuh mereka menjadi lebih optimal. Dalam proses tumbuh kembang tersebut terdapat berbagai macam kandungan dalam makanan yang dibutuhkan oleh tubuh seperti karbohidrat, vitamin, lemak, protein dan sebagainya, kandungan-kandungan itu dapat berasal dari umbi-umbian, sayur-sayuran, buah-buahan serta berasal dari hewan atau biasa disebut hewani (Rachmat *et al.*, 2016). Berdasarkan perbedaan kandungan dari tiap jenis makanan yang telah dikonsumsi maka dibutuhkan diet yang optimal dan pengetahuan gizi dalam mengatur makanan yang akan dikonsumsi agar dapat memenuhi angka kecukupan gizi untuk tiap individu. Peran diet yang optimal ini sangatlah penting khususnya bagi para atlet, karena seorang atlet memiliki kewajiban untuk selalu berlatih agar bisa mendapatkan hasil yang optimal dalam suatu perlombaan atau berprestasi. Berdasarkan kewajiban tersebut maka seorang atlet memiliki rutinitas secara fisik yang lebih berat dibandingkan mereka yang bukan atlet dan mereka yang hanya berolahraga di waktu luang. Sehingga seorang atlet harus dapat mengatur pola makan mereka terutama pada saat mendekati kejuaraan serta pada saat kejuaraan sedang berlangsung. Dalam melakukan diet yang optimal biasanya terdapat parameter berupa batasan harga terhadap komposisi bahan makanan tersebut. Pertimbangan yang dibutuhkan agar bisa mendapatkan harga termurah namun dengan komposisi terbaik untuk diet yang optimal dapat berupa harga bahan makanan dan berapa jumlah yang dibutuhkan.

Olahraga merupakan suatu kegiatan yang terdiri dari kegiatan fisik dan juga psikis seseorang yang berfungsi sebagai salah satu cara dalam merawat ataupun meningkatkan kualitas hidup seseorang (Suryandriyo, B., 2013). Terdapat empat pengelompokan olahraga berdasarkan dengan sistem kerja syaraf dan otot dalam melakukan penentuan kebutuhan energi serta zat gizi yaitu olahraga *power*, olahraga *endurance*, olahraga *sprint*, olahraga permainan. *Endurance* atau secara harfiah disebut sebagai daya tahan merupakan kemampuan yang dimiliki oleh seseorang untuk dapat bergerak yang memanfaatkan seluruh anggota tubuhnya dalam kurun waktu yang tergolong lama serta dengan tempo yang sedang hingga cepat tanpa mengakibatkan sakit ataupun kelelahan yang berat. Di dalam aktivitas olahraga, latihan *endurance* dapat dibedakan menjadi dua macam yaitu *muscular endurance (local endurance)* dan *cardiorespiratory endurance*. Olahraga menembak merupakan salah satu olahraga yang membutuhkan aspek *endurance* (ketahanan) untuk para atletnya. Namun selain

olahraga menembak, aspek *endurance* juga dibutuhkan pada olahraga lainnya seperti berenang, bersepeda, tenis dan lain sebagainya. Durasi untuk melakukan olahraga atau latihan pada atlet *endurance* memerlukan waktu yang lebih lama dibanding dengan atlet yang lainnya, untuk itu agar dapat menjadi seorang atlet *endurance* memerlukan cadangan lemak serta karbohidrat yang dapat bertahan lebih lama agar dapat diolah sebagai cadangan energi yang diperlukan untuk penggunaan pada waktu yang lama pula supaya mereka tidak kehabisan energi saat beraktivitas. Selain itu disamping olahraga yang teratur maka porsi dan komposisi makanan yang baik menjadi salah satu pendukung atlet untuk meraih kesuksesan atau berprestasi, karena diet yang optimal dan atau komposisi makanan yang baik akan menjadi sumber energi cadangan yang baik pula untuk para atlet. Sehingga untuk bisa mendapatkan jumlah karbohidrat, protein serta lemak dan kalori yang optimal maka membutuhkan perhitungan yang akurat supaya makanan yang dikonsumsi oleh para atlet *endurance* tersebut bisa memenuhi asupan gizi yang dibutuhkan dengan biaya yang minimal.

Akibat dari permasalahan tersebut maka dibutuhkanlah sebuah sistem yang dapat membantu untuk melakukan optimasi komposisi bahan makanan atlet olahraga menembak. Dalam melakukan perhitungan atau optimasi terhadap komposisi bahan makanan tersebut dapat dilakukan dengan berbagai metode, salah satunya menggunakan metode *Evolution Strategies* (ES). *Evolution Strategies* merupakan salah satu cabang dari ilmu algoritma evolusi yang sudah banyak terbukti dapat menyelesaikan permasalahan optimasi. Selain itu metode *Evolution Strategies* juga mempunyai proses eksekusi yang lebih cepat dibandingkan dengan algoritma lain misalnya Algoritma Genetika. Karakteristik utama dari *Evolution Strategies* yaitu dalam melakukan representasi kromosomnya menggunakan *vector* bilangan *real*. Serta salah satu hal yang menarik dari *Evolution Strategies* saat mengontrol nilai strategi parameter yaitu terdapat mekanisme *self-adaption* (Mahmudy,WF., 2015).

Pada penelitian sebelumnya yang telah dilakukan oleh Rahmadhani *et al.*(2016), optimasi komposisi bahan makanan atlet dapat diselesaikan dengan menggunakan Algoritma Genetika. Begitu pula dengan penelitian oleh Rachmat *et al.*(2016) dimana permasalahan optimasi komposisi bahan makanan atlet dapat pula diselesaikan dengan menggunakan metode *Particle Swarm Optimization* (PSO). Hanya saja perbedaan di antara kedua penelitian tersebut dengan penelitian penulis yaitu pada penelitian penulis saat menghitung kebutuhan gizi yang dibutuhkan oleh atlet mempertimbangkan berapa lama durasi latihan yang dilakukan oleh masing – masing atlet selain itu pembagian berapa banyak asupan gizi yang dibutuhkan (karbohidrat, protein, lemak) didasarkan oleh kebutuhan zat gizi sesuai dengan asupan yang dibutuhkan berdasarkan periodisasi diet atlet.

Pada penelitian sebelumnya yang telah dilakukan oleh Herdanis *et al.*(2016), dengan menggunakan algoritma *Evolution Strategies* terbukti dapat menyelesaikan permasalahan optimasi gizi anak panti asuhan. Selain itu di dalam penelitian yang dilakukan oleh Yansari *et al.*(2016) pun terbukti bahwa dengan

menggunakan algoritma *Evolution Strategies* dapat menyelesaikan permasalahan optimasi biaya serta asupan gizi pada pasien diet khusus.

Berdasarkan uraian tersebut maka dapat diketahui bahwa algoritma *Evolution Strategies* dapat digunakan dalam berbagai masalah yang kompleks, sehingga pada penelitian ini algoritma *Evolution Strategies* digunakan untuk melakukan optimasi komposisi bahan makanan pada atlet olahraga menembak.

1.2 Rumusan masalah

Berdasarkan dari penjelasan pada latar belakang, maka rumusan masalah pada penelitian ini adalah sebagai berikut:

1. Bagaimana menerapkan algoritma *Evolution Strategies* dalam menentukan optimasi bahan makanan atlet olahraga menembak?
2. Bagaimana pengaruh dari parameter algoritma *Evolution Strategies* yang digunakan?
3. Apakah dengan menggunakan algoritma *Evolution Strategies* sistem mampu dalam memberikan rekomendasi komposisi dengan baik?

1.3 Tujuan

Penelitian ini bertujuan untuk mengimplementasikan algoritma *Evolution Strategies* untuk dapat melakukan optimasi bahan makanan atlet olahraga menembak serta untuk mengetahui bagaimana hasil kualitas solusi dari pengimplementasian algoritma tersebut dalam melakukan optimasi bahan makanan atlet olahraga menembak.

1.4 Manfaat

Pada penelitian ini memiliki manfaat untuk membantu memberikan rekomendasi kepada para ahli agar dapat menangani atlet olahraga menembak pada saat menentukan komposisi bahan makanan yang tepat untuk para atlet. Selain itu bermanfaat untuk dapat memberikan informasi mengenai komposisi bahan makanan yang tepat untuk di konsumsi kepada para atlet olahraga menembak.

1.5 Batasan masalah

Untuk menghindari melebar nya masalah yang akan diselesaikan dalam penelitian ini, maka diberi batasan-batasan sebagai berikut:

1. Diasumsikan bahwa atlet olahraga menembak pada penelitian ini tidak memiliki alergi apapun dan dalam kondisi kesehatan yang normal.
2. Penelitian ini hanya memperhitungkan kandungan gizi yang meliputi lemak, protein dan karbohidrat.
3. Penelitian ini mempertimbangkan faktor harga sebab digunakan dalam penentuan komposisi bahan makanan untuk atlet olahraga menembak dengan biaya yang diinginkan atau mendekati biaya yang diinginkan.

4. Hasil rekomendasi komposisi bahan makanan ini diperuntukan bagi atlet yang memasuki fase periodisasi untuk *pra* kompetisi atau pertandingan (hari ke – 6, 5, 4 sebelum pertandingan).
5. Bahan makanan yang digunakan adalah berasal dari penelitian sebelumnya yaitu mengenai Optimasi Komposisi Makanan untuk Atlet Olahraga Endurance Menggunakan *Particle Swarm Optimization* oleh Zilfikri Yulfiandi Rachmat.
6. Harga pada bahan makanan sistem berasal dari salah satu pasar di daerah Malang, Jawa Timur yaitu Pasar Mergan, Jl Raya Langsep. Selain itu harga bahan makanan juga disesuaikan dengan Sistem Informasi Ketersediaan dan Perkembangan Harga Bahan Pokok di Jawa Timur (SISKAPERBAPO, 2017). Berdasarkan batasan ini maka sistem diperuntukan bagi atlet menembak sekitar Malang Kota.

1.6 Sistematika Pembahasan

Sistematika pembahasan ini bertujuan supaya pembaca dapat memahami isi dalam penulisan ini.

BAB 1 Pendahuluan

Bab ini akan membahas tentang judul serta topik yang memuat latar belakang, rumusan masalah, batasan masalah, tujuan serta manfaat penulisan dan sistematika pembahasan mengenai penelitian ini dengan judul Optimasi Komposisi Bahan Makanan Atlet Menembak dengan Menggunakan Metode *Evolution Strategies* (ES).

BAB 2 Landasan Kepustakaan

Pada bab ini akan membahas tentang dasar teori dan referensi pendukung penulisan pada penelitian Optimasi Komposisi Bahan Makanan Atlet Menembak dengan menggunakan Metode *Evolution Strategies*. Adapun teori yang telah diambil haruslah berdasarkan dari referensi – referensi yang bisa di pertanggungjawabkan, maka dari itu penulisan harus berbentuk kutipan yang bisa mengacu pada referensi tertentu.

BAB 3 Metodologi

Bab ini berisi tentang metode yang akan digunakan untuk melakukan optimasi komposisi bahan makanan atlet menembak yang terdiri dari studi literatur, pengambilan data, metode perancangan, metode implementasi, metode pengujian dan analisa hasil yang berhubungan dengan penelitian ini.

BAB 4 Perancangan Sistem

Bab berikut ini membahas mengenai perancangan sistem yang akan dibangun dengan menerapkan algoritma *Evolution Strategies* untuk optimasi komposisi bahan makanan atlet menembak.

BAB 5 Implementasi

Bab ini akan membahas mengenai pengimplementasian dari sistem optimasi komposisi bahan makanan untuk atlet menembak dengan menggunakan algoritma *Evolution Strategies* dengan tipe siklus $(\mu+\lambda)$.

BAB 6 Analisa dan Pengujian

Pada bab ini berisi mengenai hasil dari sistem yang sudah di implementasikan sebelumnya beserta hasil dari pengujian parameter dan juga analisisnya.

BAB 7 Penutupan

Bab berikut ini akan berisi mengenai kesimpulan dari perancangan, implementasi dan pengujian, serta saran untuk digunakan pada penulisan selanjutnya.



LANDASAN KEPUSTAKAAN

Pada landasan kepastakaan ini akan dibahas mengenai kajian pustaka dan dasar teori sebagai penunjang dalam pembuatan penelitian mengenai Optimasi Komposisi Bahan Makanan Atlet Olahraga Menembak dengan Menggunakan Metode *Evolution Strategies* (ES).

1.1 Kajian Pustaka

Di dalam kajian pustaka ini akan dibahas mengenai perbandingan antara penelitian ini terhadap penelitian yang sebelumnya. Pada penelitian sebelumnya yang telah dilakukan oleh Rahmadhani *et al.*(2016) menunjukan bahwa Algoritma Genetika juga dapat menyelesaikan permasalahan optimasi komposisi makanan bagi atlet *endurance*, dari penelitian ini pula dibuktikan bahwa semakin besar nilai dari probabilitas mutasi dan semakin kecilnya nilai probabilitas dari *crossover* maka akan semakin rendah nilai rata – rata *fitness*-nya karena kemampuan belajar Algoritma Genetika yang semakin lama akan menurun dari generasi yang sebelumnya serta tidak mampu dalam mengeksplorasi daerah *optimum local*. Selain itu rumus perhitungan gizi yang digunakan kurang kompleks sebab tidak mempertimbangkan durasi latihan sehingga berkemungkinan besar bahwa kebutuhan gizi yang dibutuhkan kurang dari kebutuhan yang sebenarnya.

Begitu pula dengan penelitian yang dilakukan oleh Rachmat *et al.*(2016) ditunjukan bahwa metode *Particle Swarm Optimization* juga dapat digunakan untuk menyelesaikan permasalahan tentang optimasi komposisi makanan bagi atlet *endurance*. Dari penelitian ini diketahui bahwa tingkat konvergensi serta hasilnya akan sangat bergantung kepada perkiraan awal solusi sehingga kemungkinan adanya kesamaan menu yang direkomendasikan akan semakin besar. Sama seperti penelitian yang dilakukan oleh Rahmadhani *et al.*(2016) bahwa pada penelitian ini pun dirasa masih kurang mampu dalam menetapkan angka kebutuhan gizi yang sebenarnya sehingga rekomendasi yang dihasilkan masih dirasa kurang mampu untuk memenuhi kebutuhan gizi untuk para atlet dengan durasi latihan yang beragam.

Pada dasarnya metode *Evolution Strategies* (ES) sudah mulai digunakan pada penelitian di dalam jurnal nasional maupun internasional untuk menunjang penulisan penelitian. Metode *Evolution Strategies* (ES) pernah digunakan dalam penelitian oleh Herdanis *et al.*(2016) untuk menyelesaikan permasalahan optimasi gizi anak panti asuhan. Di dalam penelitian tersebut diimplementasikan dengan merepresentasi solusi terbaik yang berupa rekomendasi hidangan dalam bentuk kromosom yang berbentuk deretan dari tiap nomor indeks masing-masing jenis makanan dalam tiap jam makan agar dapat memenuhi angka kecukupan gizi yang ideal serta biaya yang tidak melebihi anggaran serta komposisi makanan yang bervariasi untuk anak-anak panti asuhan. Permasalahan optimasi gizi pada anak panti asuhan dapat diselesaikan pada ukuran populasi 70, ukuran generasi 1000 dan ukuran C 10 serta jumlah hari 3.

Dengan parameter terbaik tersebut maka dihasilkan fitness terbaik 5.7334 dengan nilai kecukupan energi sebesar 91% AKG dengan rata-rata biaya pengeluaran perhari sebesar Rp 137.800,-

Begitu pula dengan penelitian yang dilakukan oleh Yansari *et al.*(2016) dimana metode *Evolution Strategies* digunakan dalam melakukan optimasi biaya serta asupan gizi bagi para pasien diet khusus. Dari penelitian ini pula diketahui bahwa berdasarkan uji coba terhadap 4 studi kasus maka sistem mampu memberikan hasil rekomendasi komposisi dari bahan makanan yang masih di dalam batas toleransi Angka Kecukupan Gizi (AKG) pada pasien diet khusus dengan ukuran populasi (μ) terbaik sebesar 200, *offspring* (λ) sebanyak 7 dan jumlah generasi terbaik sebanyak 250.

Supaya lebih jelas dalam memahami yang telah dipaparkan sebelumnya, maka berikut tabel ringkasan dari kajian pustaka.

Tabel 0.1 Ringkasan Perbedaan Penelitian Sebelumnya dengan Penelitian Penulis

| No | Judul | Objek | Metode | Perbandingan | |
|----|--|--|------------------------------------|---|---|
| | | | | Studi Literatur | Penelitian Penulis |
| 1 | Optimasi Komposisi Makanan Untuk Atlet <i>Endurance</i> Menggunakan Algoritma Genetika (Rahmadhani, 2016) | Atlet <i>Junior</i> Olahraga Tennis Lapangan | Algoritma Genetika | Gizi Atlet Tennis Lapangan, Komposisi Makanan, Algoritma Genetika | Gizi Atlet Menembak, Komposisi Makanan, <i>Evolution Strategies</i> |
| 2 | Optimasi Komposisi Makanan Atlet <i>Endurance</i> Menggunakan Metode <i>Particle Swarm Optimization</i> (Rachmat, 2016) | Atlet <i>Junior</i> Olahraga Tennis Lapangan | <i>Particle Swarm Optimization</i> | Gizi Atlet Tennis Lapangan, Komposisi Makanan, <i>Particle Swarm Optimization</i> | Gizi Atlet Menembak, Komposisi Makanan, <i>Evolution Strategies</i> |
| 3 | Optimasi Gizi Pada Anak Panti Asuhan Menggunakan <i>Evolution Strategies</i> (Herdanis, 2016) | Anak Panti Asuhan | <i>Evolution Strategies</i> | Gizi Anak Panti Asuhan, teknik ES ($\mu+\lambda$) | Gizi Atlet Menembak, Komposisi Makanan, teknik ES ($\mu+\lambda$) |
| 4 | Optimasi Biaya dan Asupan Gizi Pasien Diet Khusus dengan Menggunakan Algoritma <i>Evolution Strategies</i> (Yansari, 2016) | Pasien Diet Khusus | <i>Evolution Strategies</i> | Gizi Pasien Diet Khusus, teknik ES ($\mu+\lambda$) | Gizi Atlet Menembak, Komposisi Makanan, teknik ES ($\mu+\lambda$) |

Metode *Evolution Strategies* (ES) merupakan algoritma yang dapat lebih cepat dalam menemukan solusi yang mendekati dengan hasil yang optimal. Hal tersebut disebabkan karena proses generasi yang dibutuhkan juga sedikit maka proses eksekusi pada program dapat berjalan dengan cepat. Selain itu algoritma *Evolution Strategies* ini juga berbeda dengan metode optimasi lainnya seperti Algoritma Genetika dan *Particle Swarm Optimization* (PSO), sebab pada *Evolution Strategies* digunakan *self-adaption* dimana terdapat pembatasan terhadap suatu nilai mutasi di setiap generasinya. Sehingga memungkinkan dalam bereksplorasi lebih luas serta dapat lebih fokus terhadap pencapaian nilai optimum global.

Menurut Mahmudy,WF (2015), Algoritma Genetika juga dirasa kurang mampu dalam menyelesaikan permasalahan yang kompleks dengan area pencarian yang luas. Sedangkan menurut Mentari *et al.*(2014) algoritma *Particle Swarm Optimization* juga mudah terjebak dalam konvergensi dini. Maka dari itu, pada penelitian ini digunakanlah metode *Evolution Strategies* untuk optimasi komposisi bahan makanan untuk atlet menembak.

Berdasarkan dari pemaparan tersebut maka pada penelitian ini dalam menyelesaikan permasalahan optimasi komposisi bahan makanan untuk atlet menembak akan menggunakan salah satu cabang dari *Evolution Strategies* (ES) yaitu tipe proses ($\mu+\lambda$) dimana dalam proses ini menggunakan metode seleksi *elitism selection* dengan melibatkan *parent* dan *offspring* dari hasil yang didapatkan saat proses mutasi.

1.2 Dasar Teori

1.2.1 Olahraga Menembak

1.2.1.1 Definisi Olahraga Menembak

Secara umum, olahraga dapat diklasifikasikan kedalam tiga jenis yaitu olahraga *endurance*, olahraga *strength* serta campuran dari keduanya yang dapat diklasifikasikan sebagai olahraga *team* atau beregu.

Salah satu olahraga yang termasuk ke dalam jenis olahraga *endurance* adalah olahraga menembak. Olahraga menembak merupakan salah satu cabang dari olahraga prestasi. Sehingga memiliki tujuan untuk mengembangkan minat dan bakat seorang olahragawan dalam meningkatkan kedudukan bangsa (Sudarko,RA., 2009). Olahraga menembak juga merupakan jenis olahraga yang dapat membentuk karakter seseorang, karena olahraga menembak melatih seseorang untuk memiliki tingkat konsentrasi yang tinggi, dapat mengendalikan diri dan siap dalam mengambil resiko saat pengambilan keputusan secara cepat (Purwa *et al.*, 2015). Disamping itu dengan olahraga menembak ini seseorang dilatih untuk menjadi pribadi yang lebih mandiri, tekun, ulet, dan senantiasa bersemangat serta pantang menyerah.

Berdasarkan pengklasifikasian yang menyatakan bahwa olahraga menembak merupakan jenis olahraga dengan kategori olahraga *endurance* maka

olahraga menembak membutuhkan ketahanan pada fisik khususnya ketahanan otot dan *kardiorespirasi*.

1.2.1.2 Definisi Atlet Olahraga Menembak

Di dalam latihan olahraga menembak terdapat urutan – urutan yang harus dilakukan, mulai dari pemanasan dan melakukan latihan ketahanan fisik terlebih dahulu sampai pada akhirnya mereka bisa melakukan tembak presisi.

Sehingga definisi atlet olahraga menembak di dalam penelitian ini yaitu mereka yang mampu melakukan latihan olahraga menembak dengan kisaran waktu 30 menit hingga 8 jam per hari. Akibat dari kegiatan yang tersebut memiliki durasi dan sifat secara terus menerus maka seorang atlet tentunya membutuhkan asupan kalori yang cukup bukan hanya pada saat mereka berada dalam masa kompetisi melainkan pada saat masa latihan pun juga membutuhkan jumlah kalori yang seimbang dengan aktivitasnya.

Kesuksesan seorang atlet selalu ditunjang oleh dukungan para ilmuwan dibelakangnya. Kemajuan dari olahraga itu sendiri pun tidak luput dari kemajuan IPTEK sehingga agar bisa mempunyai atlet berbakat dibutuhkanlah pendekatan – pendekatan ilmiah dari berbagai disiplin ilmu khususnya yang berhubungan dengan aspek kesehatan (Apfel Glenn,C., 2011).

Maka dari itu salah satu faktor utama dari kesuksesan seorang atlet adalah harus mampu mengatur komposisi makanan yang akan dikonsumsi serta harus bisa menjaga kondisi tubuhnya agar tetap sehat.

1.2.2 Kebutuhan Gizi Atlet Olahraga Menembak

1.2.2.1 Asupan Energi pada Atlet Olahraga Menembak

Energi merupakan sumber utama yang dibutuhkan oleh tubuh sebagai asupan untuk dapat menjalankan proses metabolisme dan menjalankan aktivitas. Pada saat melakukan perhitungan mengenai Angka Kecukupan Gizi (AKG), satuan yang digunakan yaitu kilokalori (kcal) dan kalori (kal). Menurut (Supriasa dalam Suci dalam Rachmat, 2016) banyaknya energi yang didapatkan dari asupan makanan di setiap harinya harus disesuaikan dengan banyaknya energi yang dibutuhkan oleh tubuh.

Hal penting yang harus diperhatikan oleh para atlet menembak yaitu harus dapat menyeimbangkan energi yang mereka konsumsi dengan seberapa banyak energi yang akan mereka keluarkan. Durasi pada saat latihan ataupun ketika sedang melakukan kompetisi merupakan faktor yang menyebabkan kebutuhan jumlah kalori yang besar.

Apabila angka kecukupan kalori tersebut tidak terpenuhi maka bisa mengakibatkan performa yang buruk untuk atlet tersebut. Akibat Atlet Olahraga Menembak termasuk kedalam kategori Atlet Ketahanan (*endurance*), maka berikut persamaan 2.1 sampai 2.7 yang diperoleh dari Pedoman Gizi Olahraga Prestasi (Damajanti *et al.*, 2014) agar dapat digunakan dalam memperkirakan total kebutuhan energi yang dibutuhkan oleh para atlet olahraga menembak.

Tabel 0.2 Rumus Perhitungan *Basal Metabolic Rate* (BMR) berdasarkan usia dan jenis kelamin

| Rumus Perhitungan <i>Basal Metabolic Rate</i> (BMR) | |
|---|-------|
| Pria (10 – 17 tahun) : BMR = (17.5 x BB) + 651 | (2.1) |
| Pria (18 – 29 tahun) : BMR = (15.3 x BB) + 679 | (2.2) |
| Pria (30 – 60 tahun) : BMR = (11.6 x BB) + 879 | (2.3) |
| Wanita (10 – 17 tahun) : BMR = (12.2 x BB) + 746 | (2.4) |
| Wanita (18 – 29 tahun) : BMR = (14.7 x BB) + 496 | (2.5) |
| Wanita (30 – 60 tahun) : BMR = (8.7 x BB) + 829 | (2.6) |

Setelah mendapatkan hasil dari perhitungan nilai BMR tersebut maka langkah selanjutnya yaitu menghitung angka dari kebutuhan kalori yang dibutuhkan dengan menjumlahkan nilai BMR dengan SDA (10% dari nilai BMR) kemudian jumlah tersebut dikalikan dengan nilai kategori tingkatan aktivitas fisik.

$$(BMR + SDA) \times \text{aktivitas fisik} \quad (2.7)$$

Keterangan:

Basal Metabolic Rate (BMR) = Angka kebutuhan minimal kalori yang dibutuhkan tubuh (kkal).

Specific Dynamic Action (SDA) = Angka kebutuhan kalori yang dibutuhkan untuk dapat mengolah makanan dalam tubuh yaitu sebanyak 10% dari BMR (kkal).

Aktivitas Fisik = Angka kebutuhan kalori yang dibutuhkan Untuk aktivitas fisik harian (kkal).

Berat Badan (BB) = Berat tubuh seseorang dengan satuan kilogram (kg).

Angka pada tingkatan aktivitas para atlet berkaitan dengan kategorinya. Semakin tinggi intensitas kegiatan maka akan semakin tinggi pula nilai dari tingkat aktivitasnya. Berikut tabel yang menunjukkan nilai dari kategori tingkatan aktivitas fisik.

Tabel 0.3 Kategori Tingkatan Aktivitas Fisik

| Kategori | Tingkatan Aktivitas Fisik |
|--------------|---------------------------|
| Tidak Aktif | 1.2 |
| Aktif Ringan | 1.5 |
| Aktif | 1.8 |
| Sangat Aktif | 2.2 |

Sumber: Damajanti *et al.*(2014)

1.2.2.2 Pengaturan Gizi Selama Periodisasi Latihan

Periodisasi latihan merupakan perencanaan dari program latihan untuk kelompok atlet dimana perencanaan program tersebut berupa volume serta intensitas latihan yang bertujuan untuk mencegah atau meminimalisir terjadinya cedera dan juga bertujuan untuk dapat meningkatkan performa yang optimal selama periode waktu tertentu.

Periodisasi latihan itu sendiri terdiri dari tiga tahap yaitu tahap persiapan, tahap kompetisi atau pertandingan dan tahap transisi atau pemulihan (Damajanti *et al.*, 2014). Dimana pada tahap persiapan dan kompetisi atau pertandingan terdiri dari dua fase.

Pada tahap persiapan, terdapat fase persiapan umum. Fase ini akan dilakukan selama 2 sampai 3 hari bagi atlet yang memiliki kebugaran atlet yang baik namun 4 sampai 5 hari bagi atlet yang memiliki kebugaran fisik yang kurang baik. Selanjutnya akan dilakukan dengan fase persiapan khusus dimana fase ini akan dilakukan selama 2 sampai 3 minggu pertama apabila diasumsikan atlet memasuki pemusatan latihan selama satu bulan.

Pada tahap kompetisi atau pertandingan, terdapat fase *pra* kompetisi atau *pra* pertandingan. Dalam fase ini terbagi ke dalam 2 sub-fase, selain dari hari ke – 6, 5, 4 yaitu pada hari ke – 3, 2, 1 sebelum pertandingan atlet akan menjalankan metode *Carbohydrate Loading* (70% karbohidrat dari total energi). Selanjutnya pada saat pertandingan berlangsung akan memasuki fase kompetisi atau pertandingan utama.

1.2.2.3 Asupan Karbohidrat pada Atlet Olahraga Menembak

Karbohidrat merupakan zat yang memiliki fungsi sebagai sumber energi utama bagi tubuh. Dengan durasi latihan yang tergolong cukup lama serta penggunaan massa otot secara berulang – ulang maka hal tersebut menjadikan kebutuhan karbohidrat untuk para atlet olahraga menembak pun juga meningkat. Meskipun kebutuhan karbohidrat meningkat namun kenyataannya penyimpanan karbohidrat di dalam tubuh juga mempunyai batas sehingga diperlukanlah perhitungan mengenai karbohidrat agar jumlahnya sesuai dengan kebutuhan tiap – tiap atlet olahraga menembak.

Kebutuhan karbohidrat untuk para atlet menembak direkomendasikan antara 3 – 7 gram dari berat badan masing – masing individu per hari. Asupan karbohidrat tersebut haruslah diseimbangkan juga dengan kebutuhan energi yang lain (lemak dan protein) agar asupan gizinya tetap seimbang. Biasanya pada atlet ketahanan membutuhkan 60% - 65% karbohidrat dari total kalori agar bisa mendapatkan angka kebutuhan gizi yang seimbang (Damajanti *et al.*, 2014).

1.2.2.4 Asupan Protein pada Atlet Olahraga Menembak

Asupan protein yang tepat tentunya sangat diperlukan oleh para atlet olahraga menembak. Protein merupakan zat yang dibutuhkan untuk mereka sebab sebagai asupan pengganti akibat dari kenaikan pada katabolisme protein pada saat kegiatan olahraga berlangsung serta sebagai sintesis protein setelah kegiatan latihan berlangsung.

Kebutuhan protein untuk para atlet menembak direkomendasikan antara 2 – 2.2 gram dari berat badan masing – masing individu per hari. Umumnya protein tidak biasa digunakan tubuh dalam melakukan produksi energi namun menurut Damajanti *et al.* (2014), biasanya atlet membutuhkan sekitar 12% - 15% protein dari total kalori yang dibutuhkan.

1.2.2.5 Asupan Lemak pada Atlet Olahraga Menembak

Lemak memiliki peran untuk membantu dalam mengatur suhu tubuh seseorang serta untuk mengabsorpsi vitamin. Lemak dimanfaatkan pada saat latihan sedang berlangsung sebab konsentrasi dari asam lemak pada plasma sehingga peran lemak semakin dibutuhkan agar dapat membantu dalam menghemat kegunaan karbohidrat sehingga bisa membantu meningkatkan performa para atlet. Kebutuhan lemak bagi atlet olahraga menembak juga perlu diperhatikan. Kebutuhan lemak untuk para atlet menembak direkomendasikan antara 0.8 – 1.3 gram dari berat badan masing – masing individu per hari. Idealnya asupan lemak yang dibutuhkan yaitu berkisar 25% - 30% dari total kalori (Damajanti *et al.*, 2014).

1.2.2.6 Kebutuhan Gizi Seimbang

Secara umum Organisasi Kesehatan Dunia atau *World Health Organization* (WHO) menganjurkan usia remaja dan dewasa agar dapat hidup sehat untuk mengonsumsi masing – masing kebutuhan pangan dengan jumlah rata – rata sekitar 400 gr perhari. Misalnya saja saat seperti mengonsumsi sayur dan buah – buahan, bagi remaja dan orang dewasa di Indonesia dianjurkan untuk mengonsumsi sayur dan buah – buahan sebanyak 400 – 600 gr perhari (PMK RI No 41., 2014). Disamping itu pembagian porsi makan sehari – hari pun juga perlu diperhatikan agar dapat memenuhi kebutuhan gizi seimbang, dimana untuk makan pagi dibutuhkan sebanyak 25% dari angka kebutuhan energi yang dibutuhkan, makan siang sebanyak 30% sedangkan makan malam 25% (Anonim., 2011). Umumnya, indikator yang dipakai oleh para ahli dalam penilaian konsumsi pangan guna mengetahui apakah asupan yang dikonsumsi seseorang telah

terpenuhi atau belum adalah sebagai berikut, dikatakan baik apabila memiliki persentase $\geq 100\%$, dinyatakan sedang atau cukup apabila memiliki persentase $80\% - 99\%$, dinyatakan kurang apabila $70\% - 80\%$ dan dinyatakan defisit apabila persentase $< 70\%$ (Lala,M., 2011).

1.2.3 Algoritma *Evolution Strategies* (ES)

Algoritma *Evolution Strategies* (ES) adalah sebuah algoritma pencarian yang memiliki sifat probabilitas yang didasarkan oleh teori evolusi biologi. Algoritma ini juga dirancang agar dapat menyelesaikan permasalahan optimasi dengan memberikan solusi terbaik. Salah satu hal yang menarik dari *Evolution Strategies* ketika proses mutasi yaitu terdapat mekanisme *self-adaption* yang berguna dalam membatasi perubahan pada sebuah nilai pencarian (Mahmudy,WF., 2015).

Notasi yang dipergunakan pada Algoritma *Evolution Strategies* (ES) adalah

- μ (*miu*), sebagai ukuran populasi.
- λ (*lambda*), sebagai *offspring* dari hasil dari reproduksi.

1.2.3.1 Tipe Algoritma *Evolution Strategies* (ES)

Proses di dalam Algoritma *Evolution Strategies* (ES) dibagi menjadi beberapa tipe, yaitu:

a. (μ, λ)

Pada Algoritma *Evolution Strategies* (ES) dengan tipe ini maka di dalamnya tidak terdapat rekombinasi saat proses reproduksi. Selain itu proses seleksi yang digunakan adalah metode *elitism selection* dengan mengaitkan individu yang berada di dalam *offspring*.

b. $(\mu/r, \lambda)$

Menurut Mahmudy,WF (2015), tipe Algoritma *Evolution Strategies* (ES) ini dipakai dengan cara menambahkan proses rekombinasi pada saat melakukan reproduksi. Proses seleksi yang digunakan sama seperti pada tipe (μ, λ) , yaitu menggunakan metode *elitism selection*.

c. $(\mu + \lambda)$

Pada tipe ini tidak terdapat rekombinasi saat proses reproduksi. Namun proses seleksinya menggunakan metode *elitism selection* yang melibatkan *offspring* (λ) dan *parent* (μ).

d. $(\mu/r + \lambda)$

Hampir sama seperti tipe $(\mu/r, \lambda)$ yang menambahkan proses rekombinasi pada saat melakukan proses reproduksi. Namun pada saat menggunakan metode *elitism selection* pada proses seleksi melibatkan *offspring* (λ) dan *parent* (μ).

1.2.3.2 Siklus Algoritma Evolution Strategies (ES)

Pada dasarnya siklus Algoritma *Evolution Strategies* ini hampir sama seperti pada Algoritma Genetika, hanya saja di dalam algoritma ini saat menghasilkan *offspring* lebih mengutamakan mutasi.

Di dalam Algoritma *Evolution Strategies* tidak terdapat *crossover* namun terdapat proses rekombinasi yang mana proses ini bisa melibatkan lebih dari satu induk (Mahmudy,WF., 2015). Agar lebih jelas berikut gambaran mengenai siklus dari algoritma *Evolution Strategies* (ES).

```

procedure EvolutionStrategies
begin
    t = 0
    inisialisasi P(t): generate  $\mu$  individu
    while (bukan kondisi berhenti) do
        rekombinasi: produksi C(t) sebanyak  $\lambda$  dari P(t)
        mutasi C(t)
        seleksi P(t+1) dari P(t) dan C(t)
        t = t + 1
    end while
end

```

Gambar 0.1 Pseudocode Algoritma Evolution Strategies

Sumber: Mahmudy,WF (2015)

Berdasarkan gambaran dari siklus tersebut maka diketahui bahwa siklus di dalam Algoritma *Evolution Strategies* dibagi menjadi beberapa tahap, yaitu:

- **Representasi Kromosom**

Di dalam representasi kromosom pada Algoritma *Evolution Strategies* memiliki bentuk yang *real code*. Selain itu pada algoritma ini dikenal juga *sigma* (σ) yang memiliki sifat adaptif atau bisa berubah sampai proses generasi selesai (Mahmudy,WF., 2015).

- **Generate Populasi Awal**

Menurut Mahmudy,WF (2015) inisialisasi populasi awal di lambangkan dengan *miu* (μ) dan dilakukan secara random. Contoh dari populasi awal dapat dilihat pada tabel berikut ini.

Tabel 0.4 Contoh generate Populasi Awal

| P(t) | x1 | x2 | σ_1 | σ_2 | f(x1,x2) |
|----------------|----------|---------|------------|------------|----------|
| P ₁ | 1.48980 | 2.09440 | 0.14197 | 0.91090 | 19.82128 |
| P ₂ | 8.49170 | 2.57540 | 0.53801 | 0.86904 | 34.70609 |
| P ₃ | -1.84610 | 1.70970 | 0.99835 | 0.49351 | 11.58639 |
| P ₄ | 5.81140 | 5.07790 | 0.40521 | 0.98911 | 14.56208 |

Sumber: Mahmudy,WF (2015)

- **Reproduksi**

Tujuan dari reproduksi ini adalah agar dapat memberikan hasil keturunan dari individu – individu yang ada pada populasi. Berbeda dengan Algoritma Genetika, dimana pada Algoritma Genetika menggunakan dua operator yaitu *crossover* dan mutasi, sedangkan di dalam algoritma *Evolution Strategies* menggunakan rekombinasi dan mutasi (Mahmudy,WF., 2015).

- **Rekombinasi**

Pada dasarnya Rekombinasi bukan merupakan proses utama pada algoritma *Evolution Strategies* (ES). Sebab tidak semua tipe dalam Algoritma *Evolution Strategies* menggunakan proses rekombinasi di dalamnya. Cara melakukan proses rekombinasi yang mudah yaitu menerapkan rata – rata dari masing – masing elemen induk (Mahmudy,WF., 2015). Contoh dari rekombinasi dapat dilihat pada tabel berikut.

Tabel 0.5 Contoh Rekombinasi

| C(t) | Induk | x1 | x2 | σ_1 | σ_2 | f(x1,x2) |
|----------------|-----------|----------|---------|------------|------------|----------|
| C ₁ | P1 dan P3 | -0.17815 | 1.90205 | 0.57016 | 0.70221 | 16.64183 |
| C ₂ | P2 dan P3 | 3.3228 | 2.14255 | 0.76818 | 0.68128 | 19.5813 |
| C ₃ | P1 dan P4 | 3.6506 | 3.58615 | 0.27359 | 0.95001 | 9.57005 |
| C ₄ | P2 dan P4 | 7.15155 | 3.82665 | 0.47161 | 0.92907 | 12.52404 |
| C ₅ | P1 dan P3 | -0.17815 | 1.90205 | 0.57016 | 0.70221 | 16.64183 |
| C ₆ | P3 dan P4 | 1.98265 | 1.98265 | 1.70178 | 0.74131 | 12.65007 |

Sumber: Mahmudy,WF (2015)

- **Mutasi**

Mutasi adalah tahapan proses yang paling diandalkan pada Algoritma *Evolution Strategies* (ES). Teknik dalam melakukan mutasi yang sering digunakan menurut Mahmudy,WF (2015) dapat dilihat pada persamaan dibawah ini.

$$x' = x + \sigma N(0,1) \quad (2.8)$$

Persamaan tersebut dapat didetailkan sebagai berikut:

$$x'_1 = x_1 + \sigma_1 N(0,1) \quad (2.9)$$

$$x'_2 = x_2 + \sigma_2 N(0,1) \quad (2.10)$$

Keterangan:

x = sesuai rentang dari *random* bilangan *real* yang telah ditentukan

σ = secara *random* bernilai antara 0 – 1

$$N(0,1) = \sqrt{-2 \cdot \ln r_1} * \sin 2\pi r_2 \quad (2.11)$$

Keterangan:

N(0,1)= bilangan *random* yang mengikuti sebaran normal dengan nilai 0 – 1.

Untuk mendapatkan nilai N(0,1) maka nilai r_1 dan r_2 dibangkitkan secara random pada interval [0,1]. Nilai dari σ dapat ditingkatkan apabila nilai *fitness*

dari setidaknya 1 *offspring* lebih baik dibandingkan dengan *parent* nya, dengan rumusan $\sigma' = \sigma \times 1,1$. Namun apabila tidak maka nilai σ diturunkan $\sigma' = \sigma \times 0,9$ (Mahmudy,WF., 2015). Berikut contoh dari hasil mutasi berdasarkan hasil rekombinasi yang dilakukan.

Tabel 0.6 Contoh Mutasi Berdasarkan Rekombinasi Sebelumnya

| C(t) | N ₁ (0.1) | N ₂ (0.1) | X' ₁ | X' ₂ | $\sigma'1$ | $\sigma'2$ | f(x ₁ ,x ₂) |
|----------------|----------------------|----------------------|-----------------|-----------------|------------|------------|------------------------------------|
| C ₁ | 0.1885 | 0.2747 | -0.07068 | 2.09495 | 0.62717 | 0.77243 | 21.33870 |
| C ₂ | -1.7947 | -0.1359 | 1.94415 | 2.04997 | 0.84499 | 0.74940 | 19.90344 |
| C ₃ | -0.5603 | -1.8657 | 3.49731 | 1.81372 | 0.30095 | 1.04501 | 10.98098 |
| C ₄ | 0.6189 | -0.4613 | 7.44343 | 3.39807 | 0.42445 | 0.83617 | 5.40751 |
| C ₅ | -0.1371 | -0.4201 | -0.25632 | 1.60705 | 0.51314 | 0.63199 | 11.26206 |
| C ₆ | 1.1125 | -0.2153 | 2.76338 | 3.23420 | 0.77195 | 0.81544 | 16.32937 |

Sumber: Mahmudy,WF (2015)

• Fitness

Nilai *fitness* merupakan suatu nilai yang menyatakan apakah solusi atau individu yang dihasilkan sudah baik atau buruk, selain itu nilai ini juga akan dijadikan sebagai acuan agar bisa mencapai nilai yang optimal di dalam algoritma *Evolution Strategies* (Yansari *et al.*, 2016).

Fungsi *fitness* di dalam penelitian ini bertujuan agar dapat meminimalkan fungsi *h* serta memaksimalkan nilai *f*, sehingga secara umum nilai dari fungsi *fitness*-nya yaitu

$$f = \frac{1}{(h+a)} \quad (2.12)$$

Ket:

f = *fitness*

h = fungsi yang diminimalkan

a = bilangan konstanta sesuai dengan permasalahan

Di dalam penelitian ini pada saat menentukan nilai *fitness*, terdapat parameter – parameter yang dapat memberikan pengaruh terhadap hasil perhitungan yaitu nilai penalti. Nilai dari total penalti tersebut terdiri atas penalti gizi serta penalti harga sebab penelitian ini bertujuan agar bisa meminimalkan harga namun dengan memperhatikan nilai dari penalty (Yansari *et al.*, 2016).

Penalti Gizi

Pada saat melakukan perhitungan penalti gizi maka langkah pertama yang dilakukan yaitu menghitung berapa berat bahan makanan sebenarnya berdasarkan dari nilai yang terdapat pada gen kromosomnya. Untuk dapat menghitung berat bahan makanan yang sebenarnya maka peneliti akan mengambil angka 400 g sebagai acuan dari rata – rata berat bahan makanan per hari.

$$berat = gen \times 400 g \quad (2.13)$$

Dimana dari persamaan tersebut diketahui bahwa berat merepresentasikan berat makanan yang sebenarnya dan gen merupakan nilai dari gen-nya. Apabila nilai berat bahan makanan sebenarnya telah diketahui maka selanjutnya adalah menghitung berapa kandungan gizi dari bahan makanan tersebut.

$$\text{Jumlah Kandungan Energi} = \frac{berat}{100} \times \text{nilai gizi Energi}_i \quad (2.14)$$

$$\text{Jumlah Kandungan Karbohidrat} = \frac{berat}{100} \times \text{nilai gizi Karbo}_i \quad (2.15)$$

$$\text{Jumlah Kandungan Lemak} = \frac{berat}{100} \times \text{nilai gizi Lemak}_i \quad (2.16)$$

$$\text{Jumlah Kandungan Protein} = \frac{berat}{100} \times \text{nilai gizi Protein}_i \quad (2.17)$$

Dimana dari persamaan tersebut untuk masing – masing nilai gizi pada energi, karbohidrat, lemak dan protein merupakan nilai gizi dari bahan makanan tersebut per 100 g.

Apabila telah diketahui jumlah kandungan gizi setiap bahan makanan maka selanjutnya adalah melakukan perhitungan nilai penalti gizi untuk masing – masing jenis bahan makanannya dengan pembobotan jenis bahan makanan tersebut.

$$\begin{aligned} \text{penaltiMenu}_i = & |(\text{Jumlah Kandungan Energi}_i - (\text{Kebutuhan Energi} * \\ & \text{BPM}))| + |(\text{Jumlah Kandungan Karbohidrat}_i - (\text{Kebutuhan Karbohidrat} \\ & * \text{BPM}))| + |(\text{Jumlah Kandungan Lemak}_i - (\text{Kebutuhan Lemak} * \text{BPM}))| \\ & + |(\text{Jumlah Kandungan Protein}_i - (\text{Kebutuhan Protein} * \text{BPM}))| \end{aligned} \quad (2.18)$$

Dimana dari persamaan tersebut diketahui Bobot Prioritas Menu (BPM) yang telah ditentukan berdasarkan pembagian porsi makan sehari – hari guna mendapatkan kebutuhan gizi seimbang.

Apabila telah didapatkan nilai dari hasil perhitungan penalti untuk tiap jenis bahan makanan maka selanjutnya adalah menghitung nilai dari total penaltinya.

$$\text{Total Penalty Menu} = \text{Penalti Menu Pagi} + \text{Penalti Menu Siang} + \text{Penalti Menu Malam} \quad (2.19)$$

Penalti Harga

Pada dasarnya sama seperti penalti gizi yaitu memberikan bobot pada saat suatu individu membuat suatu pelanggaran, penalti harga dilakukan supaya dapat memberikan bobot apabila total harga yang direkomendasikan melebihi dari anggaran yang telah ditentukan.

$$\text{total Harga} = \sum_{i=0}^n \frac{berat}{100} \times \text{harga}_i \quad (2.20)$$

Berdasarkan dari persamaan tersebut diketahui bahwa i merupakan gen ke-, sedangkan n adalah total jumlah gennya. Kemudian harga yang digunakan adalah harga bahan makanan untuk setiap 100 gr.

$$\text{penalti} \begin{cases} 0 & \text{total Harga} - \text{anggaran} < 0 \\ \text{total harga} - \text{anggaran} & \text{total Harga} - \text{anggaran} > 0 \end{cases} \quad (2.21)$$

Berdasarkan dari persamaan tersebut, nilai total harga merupakan total harga dari bahan makanan pada suatu kromosom sedangkan anggaran merepresentasikan berapa banyak anggaran yang diinginkan oleh user.

- **Seleksi**

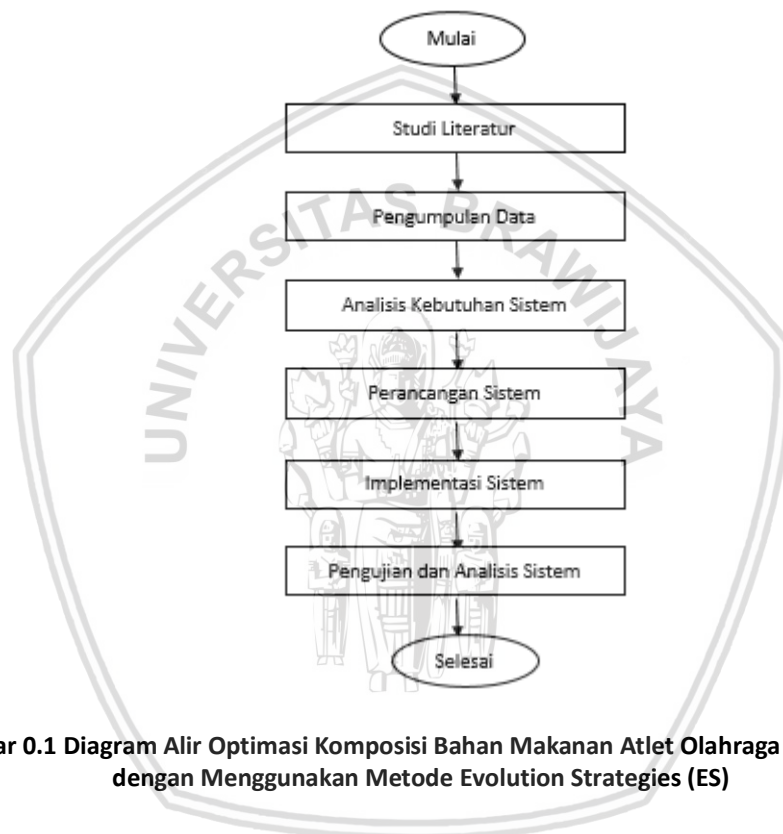
Umumnya proses seleksi yang digunakan yaitu terdiri dari 2 macam. Yang pertama proses tanpa melibatkan individu induk dan proses dengan melibatkan individu induk yang bergantung dari tipe Algoritma *Evolution Strategies* yang digunakan. Seleksi ini dapat dikatakan juga sebagai proses *elitsm selection*. Apabila nilai *fitness* kromosom tersebut besar maka akan semakin tinggi pula peluang kromosom tersebut terpilih supaya dapat terbentuk generasi selanjutnya yang lebih baik dibandingkan dengan generasi yang sebelumnya, sehingga proses *elitsm selection* ini akan menjamin individu yang terbaiklah yang akan selalu lolos (Mahmudy, WF., 2015).



METODOLOGI

Di dalam bab metodologi ini akan dibahas mengenai langkah – langkah serta rancangan yang akan digunakan dalam pembuatan sistem Optimasi Komposisi Bahan Makanan Atlet Olahraga Menembak Dengan Menggunakan Metode *Evolution Strategies* (ES).

Pada gambar berikut merupakan alur yang menjelaskan mengenai metodologi yang akan digunakan.



Gambar 0.1 Diagram Alir Optimasi Komposisi Bahan Makanan Atlet Olahraga Menembak dengan Menggunakan Metode Evolution Strategies (ES)

1.1 Studi Literatur

Pada tahap studi literatur maka hal yang dilakukan adalah pengumpulan dasar teori dan mempelajarinya sebagai penunjang dalam penulisan skripsi, dasar-dasar teori tersebut dapat berasal dari buku, jurnal, dan penelitian sebelumnya yang menunjang penulisan skripsi ini. Referensi utama yang menunjang dalam penulisan ini adalah metode *Evolution Strategies* dan tentang olahraga menembak serta kandungan gizi tiap bahan makanan dan kebutuhan gizi atlet olahraga menembak berdasarkan jenis kelamin, umur, berat badan dan faktor aktivitas serta berapa lama durasi latihan tiap individu.

1.2 Pengumpulan Data

Pengambilan data sebagai penunjang skripsi ini adalah dengan melakukan survey di UKM Menembak BASIC SC Universitas Brawijaya dan atau PERBAKIN bila diperlukan untuk mendapatkan data yang berupa sampel individu sebanyak 24 responden. Selain itu penulis juga menggunakan data bahan makanan berdasarkan penelitian sebelumnya oleh Rachmat *et al.*(2016) mengenai optimasi komposisi makanan untuk atlet *endurance* sebanyak 125 bahan makanan. Data – data yang digunakan dalam penelitian ini nantinya akan digunakan dalam perhitungan nilai *fitness* terbaik sehingga menghasilkan komposisi bahan makanan untuk atlet olahraga menembak yang optimal.

1.3 Analisis Kebutuhan Sistem

Supaya seluruh kebutuhan minimal dalam sistem Optimasi Komposisi Bahan Makanan Atlet Olahraga Menembak Dengan Metode *Evolution Strategies* (ES) ini dapat diimplementasikan maka dibutuhkanlah analisis kebutuhan sistem. Analisis ini disesuaikan dengan kebutuhan *hardware* dan *software* sebagai berikut ini.

Kebutuhan Hardware, meliputi:

- *Laptop dengan memory 8GB*

Kebutuhan Software, meliputi:

- *Microsoft windows 10 sebagai sistem operasi*
- *Minimum Netbeans 7.3.1 sebagai pembuat sistem berbasis desktop dengan Bahasa pemrograman java*
- *Kebutuhan data yang terdiri dari,*
 - *Data bahan makanan beserta kandungan gizinya*
 - *Data kebutuhan gizi berdasarkan umur, berat badan, jenis kelamin, faktor aktivitas dan berapa lama durasi latihan tiap individu*
 - *Daftar harga bahan makanan*

1.4 Perancangan Sistem

Perancangan sistem ini berisi mengenai formulasi permasalahan, siklus dari *Evolution Strategies*, siklus dari penyelesaian permasalahan mengenai perhitungan manual, perancangan pengujian serta perancangan antarmuka pada implementasi sistem.

1.5 Pengujian dan Analisis Sistem

Pengujian sistem ini berfungsi agar bisa mendapatkan hasil dari parameter solusi terbaik pada penelitian ini. Sedangkan analisa sistem berguna agar bisa mendapatkan hasil pengujian untuk mendapatkan solusi yang optimal.

Diharapkan dari pengujian – pengujian tersebut bisa didapatkan hasil dari parameter terbaik di setiap pengujian.

1.6 Kesimpulan

Kesimpulan ini berisi mengenai jawaban dari semua rumusan masalah yang telah dibuat, sehingga tujuan dari penelitian dapat direalisasikan. Kesimpulan ini dibuat berdasarkan dari hasil pengujian serta analisis sistem.



PERANCANGAN SISTEM

Di dalam bab perancangan sistem ini akan dijelaskan mengenai formulasi permasalahan dan perhitungan manual serta rancangan interface dan pengujian sistem.

1.1 Formulasi Permasalahan

Di dalam bagian formulasi permasalahan ini akan dibahas mengenai perancangan algoritma agar dapat memperoleh solusi *output* yang terbaik. Langkah dari proses ini diawali dengan memperhitungkan jumlah kalori serta menentukan kebutuhan karbohidrat, protein dan lemak dari tiap – tiap user, kemudian masuk ke dalam tahapan proses yang ada di metode *Evolution Strategies*. Setelah itu akan dibahas pula mengenai perancangan user interface sistem yang akan dibuat.

Agar bisa menyelesaikan permasalahan Optimasi Komposisi Bahan Makanan Atlet Olahraga Menembak Menggunakan Metode *Evolution Strategies* (ES) terdapat beberapa proses yang terdiri dari:

1. Menghitung kebutuhan zat gizi (karbohidrat, protein dan lemak) dengan menginputkan data berupa:
 - Jenis kelamin
 - Usia
 - Berat badan
 - Durasi latihan
 - Tingkat aktivitas
 - Anggaran yang diinginkan
 - Bahan makanan yang diinginkan
2. Menginisialisasi populasi awal, yaitu:
 - Jumlah populasi (μ)
 - Ukuran *offspring* (λ)
 - Jumlah generasi
3. Membuat populasi awal dimana populasi ini sesuai dengan jumlah yang telah diinputkan.
4. Membuat populasi baru untuk membentuk *offspring* dengan menggunakan mutasi, yang mana hasilnya akan diikutkan dalam proses seleksi dengan bersama *parent*
5. Melakukan proses perhitungan nilai *fitness* dari tiap individu yang kemudian akan dilakukan proses seleksi.

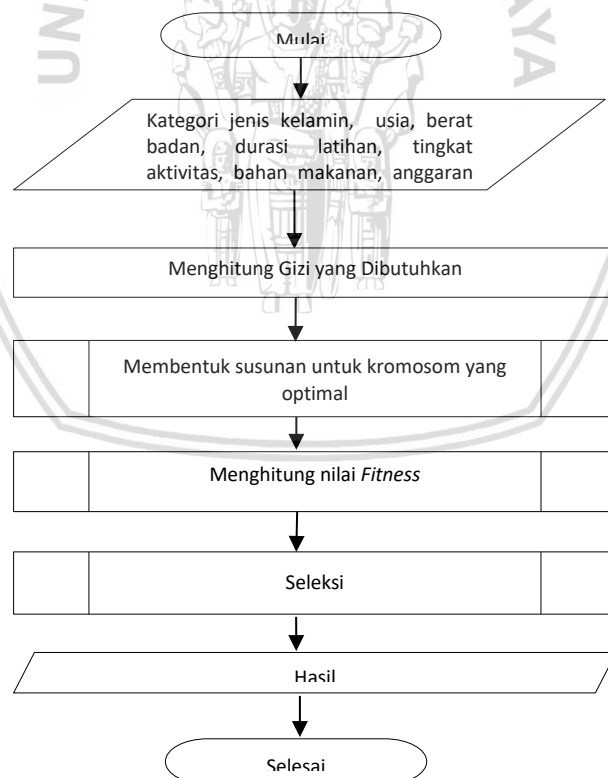
6. Metode *elitism selection* digunakan pada proses seleksi agar bisa mendapatkan hasil individu yang akan bertahan pada generasi berikutnya.

Apabila kondisi akhir telah terpenuhi maka proses akan selesai. Hasil akhir dari proses ini yaitu kromosom terbaik yang ada pada akhir generasi.

1.2 Siklus Penyelesaian Masalah

Pada siklus penyelesaian masalah ini akan dibahas mengenai proses dari perancangan sistem agar dapat mendapatkan solusi output terbaik. Langkah ini diawali dengan melakukan penentuan jenis kelamin, usia, berat badan, durasi latihan dan tingkat aktivitas yang kemudian akan dilanjutkan dengan melakukan *generate* populasi awal.

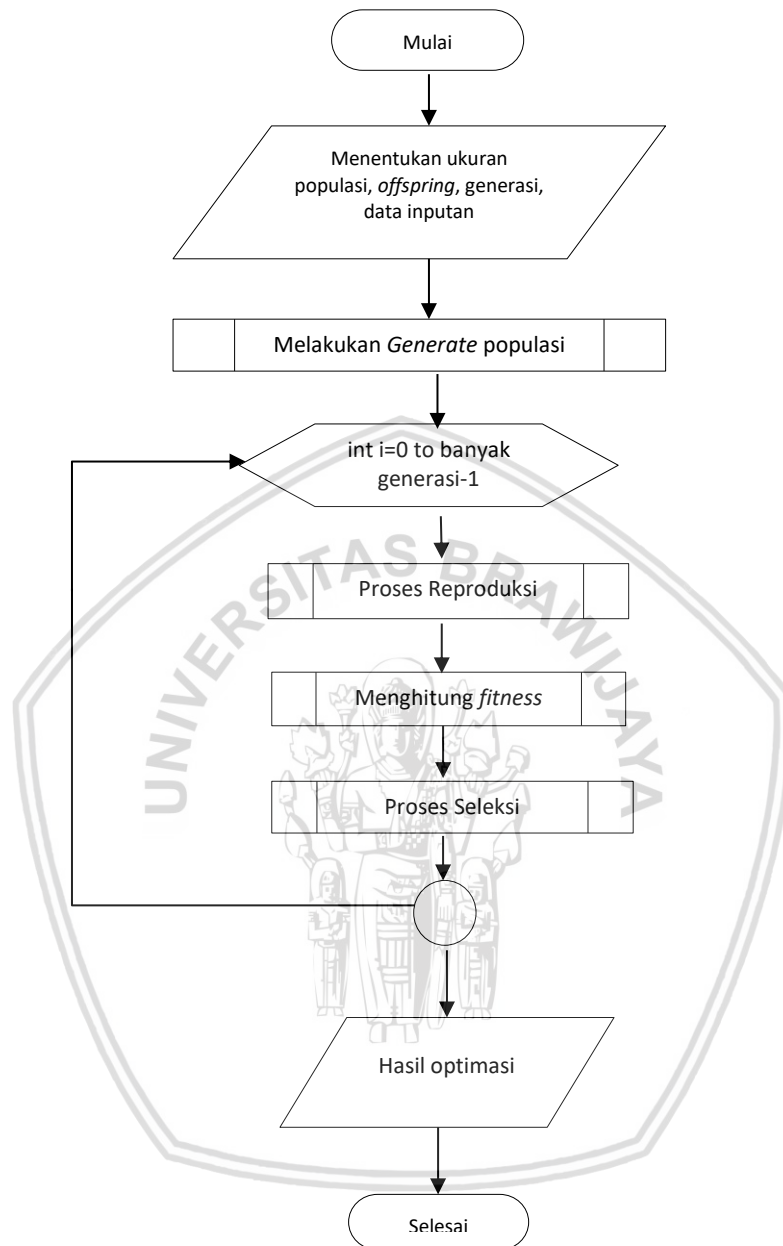
Selanjutnya *offspring* yang akan dihasilkan berasal dari jumlah kombinasi induk yang kemudian individu – individu yang berasal dari populasi awal dan *offspring* pada mutasi akan digabungkan saat proses seleksi. Proses seleksi ini dilakukan dengan menghitung *fitness* dari tiap individu. Apabila individu tersebut mempunyai *fitness* terbaik setelah n generasi maka individu tersebut akan bertahan. Untuk nilai *fitness* itu sendiri didapatkan dari nilai penalti harga serta penalti gizi yang didapatkan.



Gambar 0.1 Flowchart Optimasi Gizi Atlet Menembak

Berdasarkan Gambar 4.1 maka diketahui bahwa terdapat subproses untuk melakukan pembentukan kromosom yang optimal. Subproses tersebut dapat dilihat pada Gambar 4.2.

1.2.1 Penyusunan Kromosom Optimal

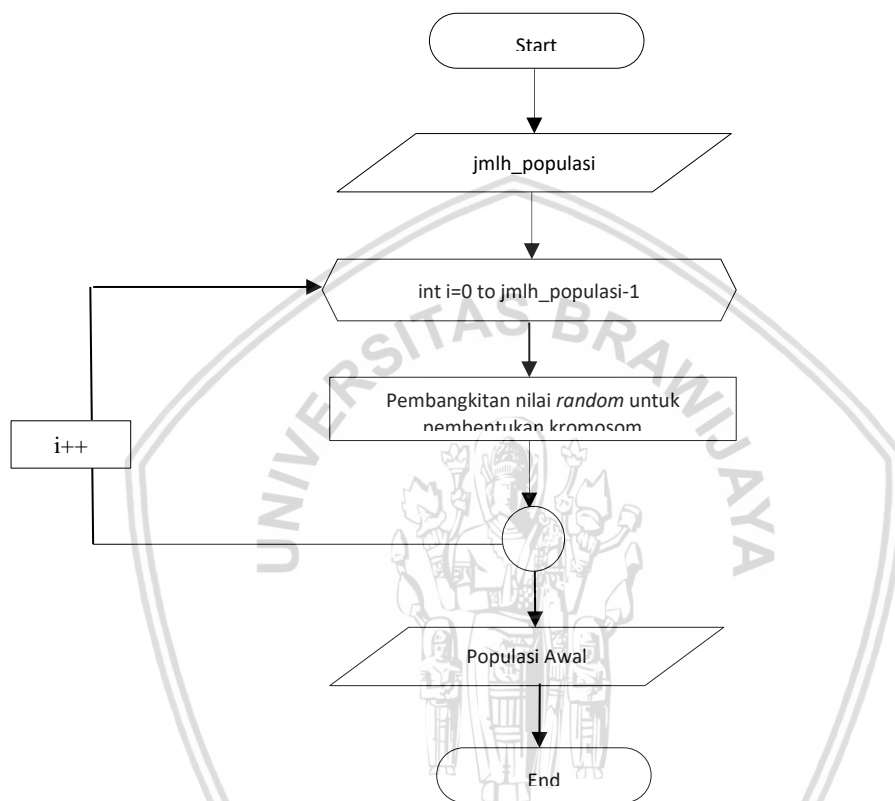


Gambar 0.2 Flowchart Proses Penyusunan Kromosom Optimal

Berdasarkan Gambar 4.2 maka diketahui dalam menyelesaikan permasalahan optimasi komposisi bahan makanan dengan menggunakan algoritma *Evolution Strategies* berdasarkan kebutuhan gizi dari masing – masing atlet terdapat empat proses yaitu diawali dengan melakukan generate populasi awal, proses reproduksi, menghitung nilai *fitness* dan terakhir melakukan proses seleksi untuk populasi baru dengan cara *elitism selection*. Pengulangan proses akan dilakukan hingga banyak generasi tercapai sampai pada akhirnya akan didapatkan *output* dengan hasil yang optimal.

1.2.2 Generate Populasi Awal

Di dalam proses melakukan pembentukan populasi awal, akan dilakukan dengan menentukan populasi secara *random* dimana populasi tersebut terdiri dari sejumlah individu yang telah diinputkan oleh *user*. Representasi kromosom akan terbentuk dari masing – masing individu tersebut dengan menggunakan bilangan *real* dimana bilangan tersebut merepresentasikan bobot tiap bahan makanan.



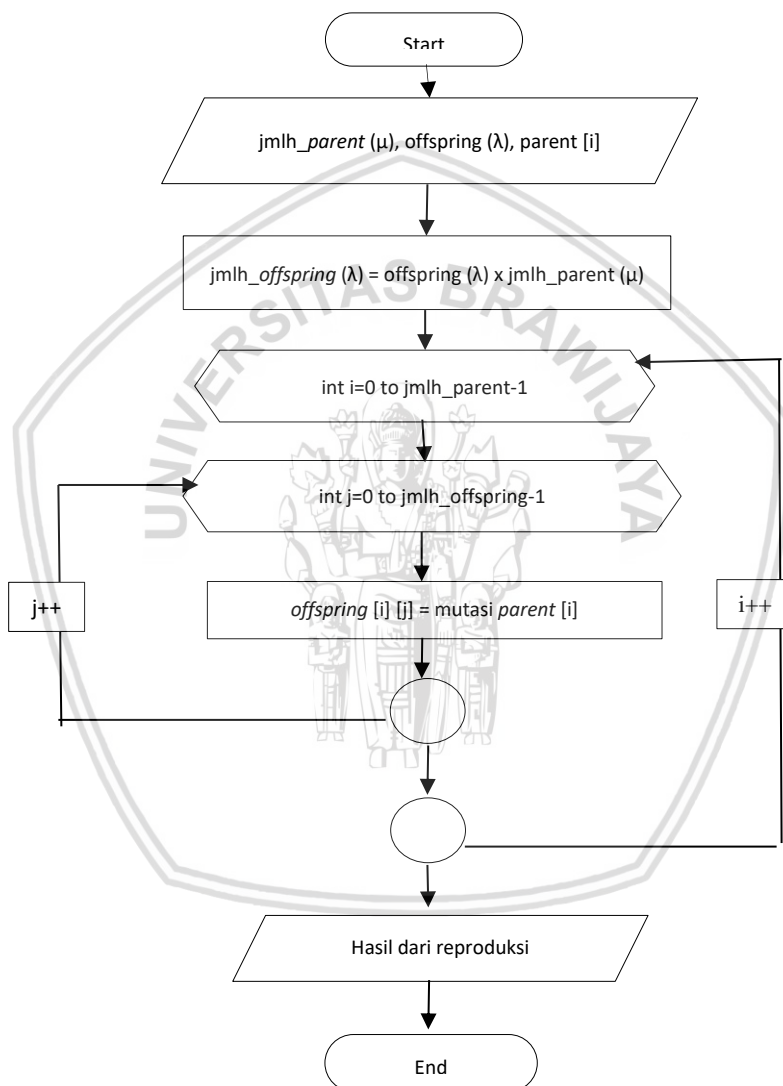
Gambar 0.3 Flowchart Proses Generate Populasi Awal

Berdasarkan Gambar 4.3 diatas maka diketahui langkah pembentukan populasi awal sebagai berikut ini:

1. Melakukan penentuan panjang dari kromosom sesuai dengan banyak total bahan makanan.
2. Melakukan pengisian masing – masing gen yang terdapat pada kromosom dengan membangkitkan nilai *random* [0..1] yang mana bilangan *random* tersebut akan merepresentasikan suatu berat pada bahan makanan.
3. Selanjutnya sistem akan menerima *input*-an berapa jumlah populasi yang akan dibentuk yang kemudian akan diproses untuk pembentukan individu.
4. Hasil dari proses ini akan di dapatkan suatu populasi awal dimana populasi tersebut akan digunakan untuk melakukan proses dari algoritma *Evolution Strategies* yang selanjutnya.

1.2.3 Reproduksi

Di dalam melakukan proses reproduksi ini maka sistem akan melakukan pembentukan banyaknya λ (λ) yang mana menyatakan banyaknya *offspring* yang nantinya akan dikalikan dengan banyaknya pada saat proses mutasi. Dimana hasil dari mutasi tersebut akan diikuti bersama dengan *parent* saat melakukan proses seleksi. Di dalam algoritma *Evolution Strategies*, proses reproduksi melibatkan *self adaption* yaitu nilai σ yang berasal dari *offspring* maka akan berubah sesuai dengan nilai dari *fitness* yang dihasilkan.



Gambar 0.4 Flowchart Pada Proses Reproduksi

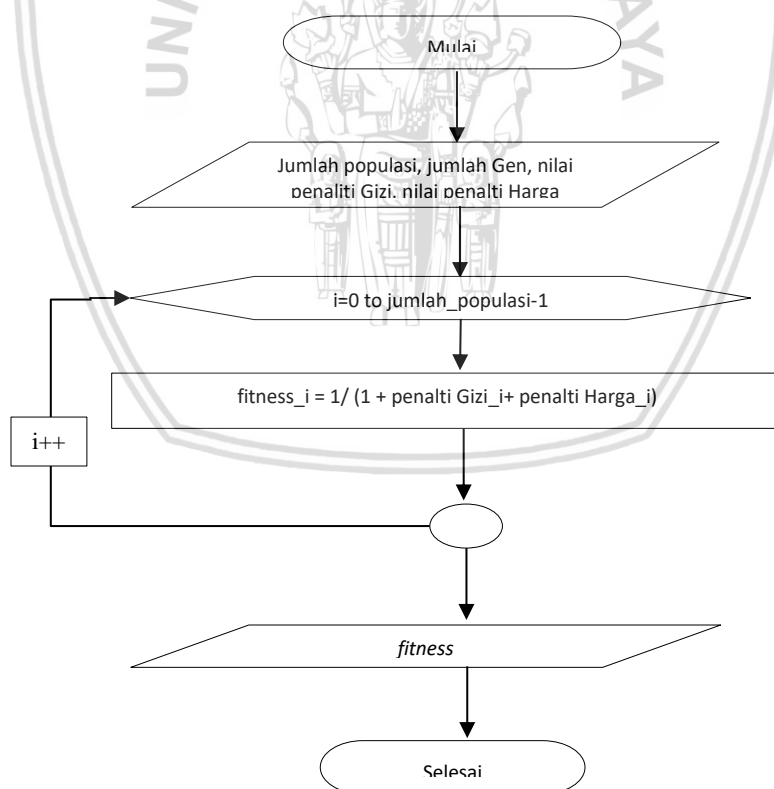
Berdasarkan dari Gambar 4.4 maka diketahui langkah pada saat melakukan proses reproduksi adalah sebagai berikut ini:

1. *Parent* yang digunakan saat proses mutasi merupakan *input*-an saat melakukan proses reproduksi.
2. Sistem melakukan *looping* sesuai dengan jumlah dari *parent* agar bisa melakukan pembentukan *offspring*.

3. Jumlah anak (*child*) akan menentukan seberapa banyak proses mutasi dalam melakukan pembentukan *offspring*.
4. Apabila proses dari mutasi telah selesai maka sistem akan mengganti nilai untuk *strategy parameters* pada *parent* asal dimana nilai tersebut akan sesuai dengan nilai dari *fitness* hasil *offspring* agar dapat melakukan proses *self adaption*.
5. Hasil dari proses mutasi ini berupa nilai *offspring* dimana nilai tersebut akan diseleksi pada saat melakukan proses selanjutnya.
6. Setelah itu maka akan dilakukan evaluasi dengan mengambil individu – individu terbaik sesuai dengan banyaknya populasi awal berdasarkan urutan dari nilai *fitness* populasi gabungan yang diurutkan secara *descending* (besar ke kecil).

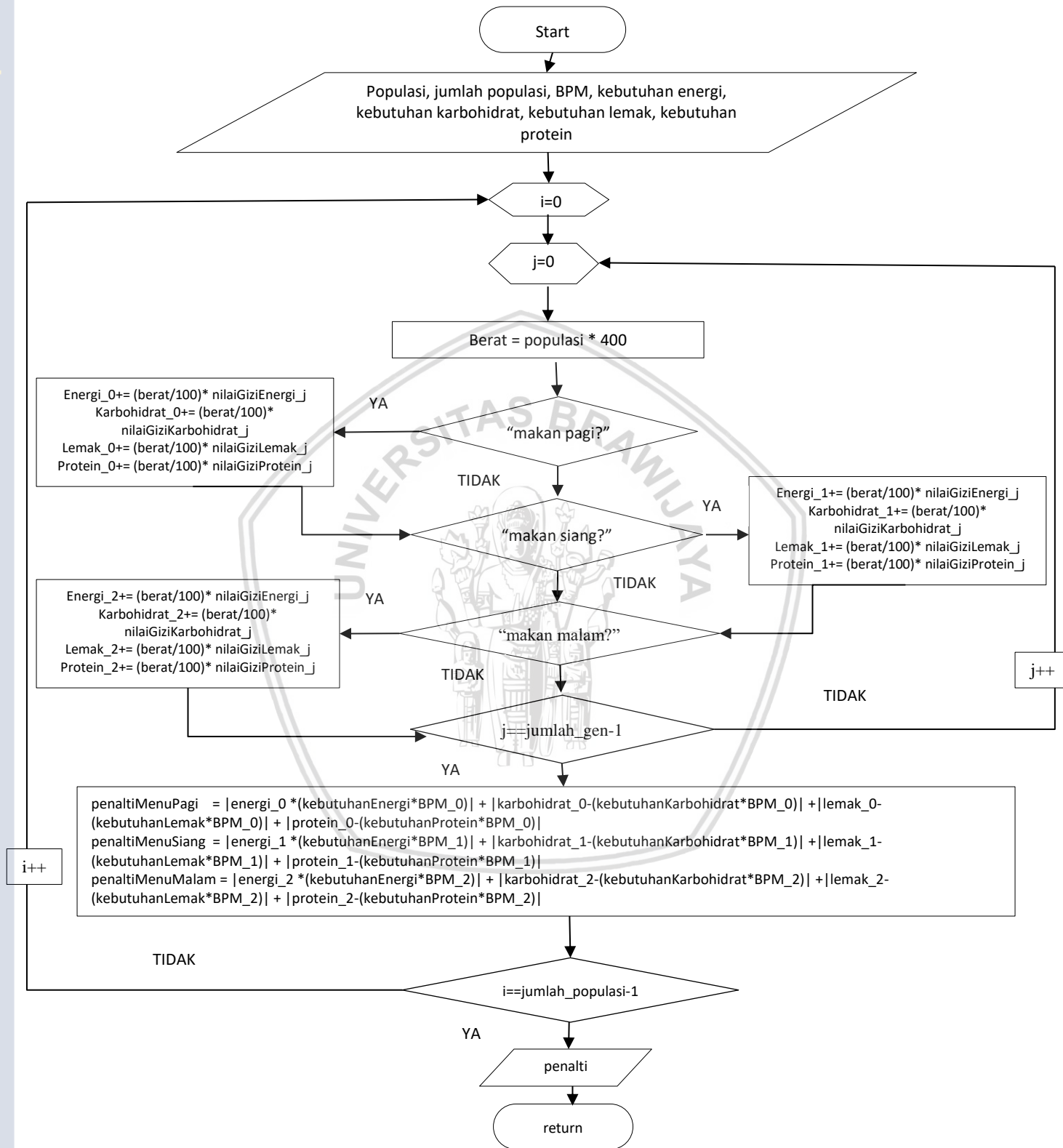
1.2.4 Nilai *fitness*

Proses perhitung nilai *fitness* perlu dilakukan sebelum melakukan proses seleksi. Perhitungan nilai *fitness* ini pun dilakukan untuk seluruh individu, baik itu adalah *parent* maupun *offspring*.



Gambar 0.5 Flowchart Proses Menghitung Nilai *fitness*

1.2.5 Penalti Gizi

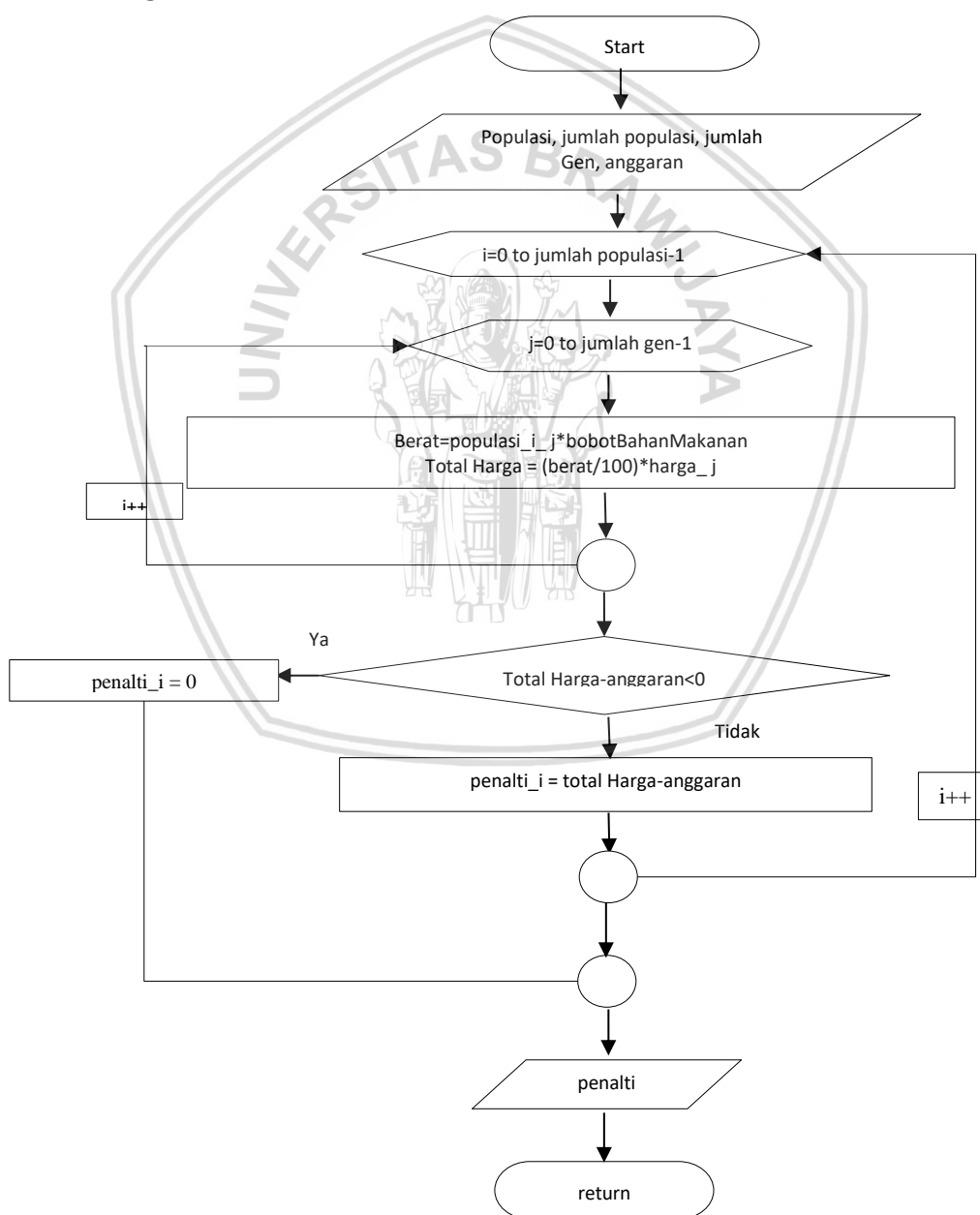


Gambar 0.6 Flowchart Proses Hitung Nilai Penalti Gizi

Sehingga diketahui langkah – langkah untuk dapat menghitung berapa nilai dari penalti gizi adalah sebagai berikut ini:

1. Melakukan perhitungan berapa berat bahan makanan yang sebenarnya sesuai dengan nilai gen menggunakan persamaan (2.13).
2. Melakukan perhitungan nilai gizi dengan menggunakan persamaan (2.14) sampai dengan persamaan (2.17).
3. Apabila telah mendapatkan masing – masing nilai gizi maka hitung nilai penalti tiap jenis bahan makanan menggunakan persamaan pada (2.18).
4. Menghitung nilai total penalti menu menggunakan persamaan (2.19).

1.2.6 Penalti Harga



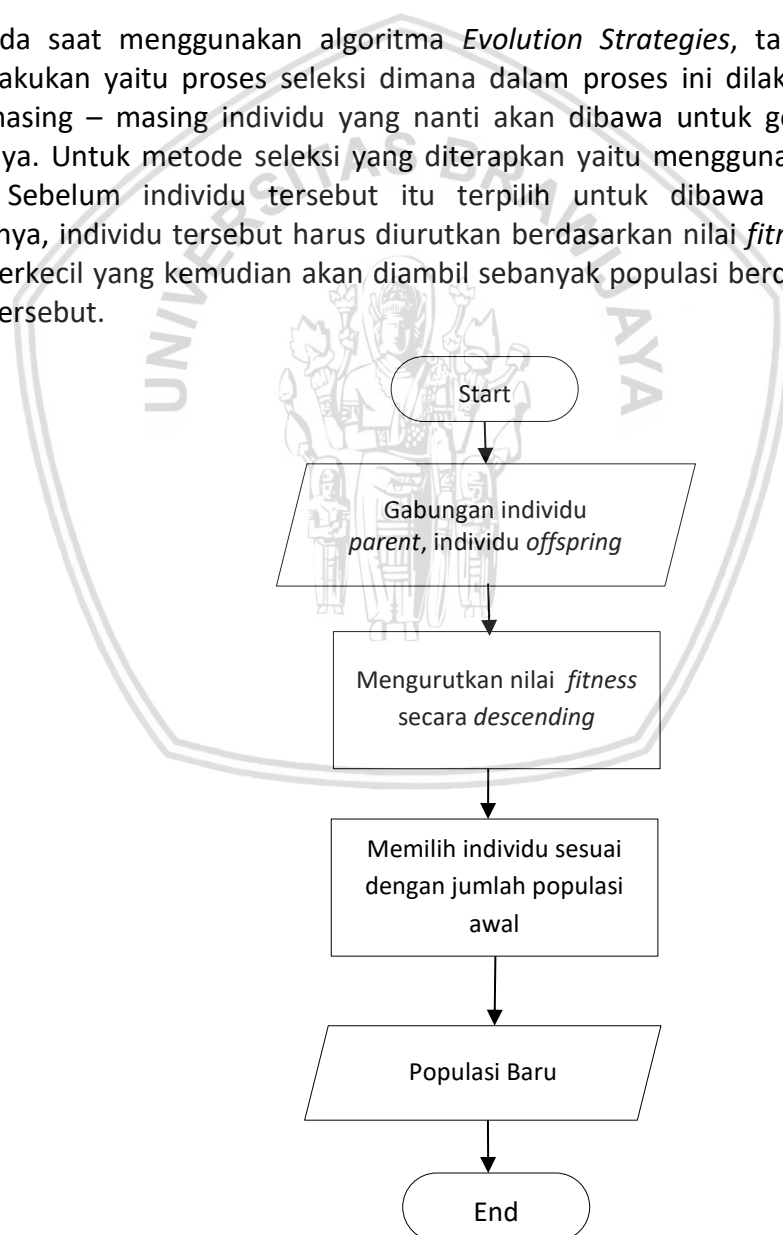
Gambar 0.7 Flowchart Proses Melakukan Perhitungan Nilai Penalti Harga

Sehingga langkah – langkah untuk dapat menghitung berapa nilai dari penalti gizi adalah sebagai berikut ini:

1. Melakukan perhitungan berapa berat bahan makanan yang sebenarnya yang di dasarkan oleh nilai gennya dengan menggunakan persamaan pada (2.13).
2. Kemudian dilanjutkan dengan melakukan perhitungan total harga yang di dasarkan oleh berat bahan makanan yang sebenarnya yang telah didapatkan sebelumnya dengan menggunakan persamaan (2.20).
3. Selanjutnya diikuti dengan melakukan perhitungan nilai penalti dengan menggunakan persamaan (2.21).

1.2.7 Seleksi

Pada saat menggunakan algoritma *Evolution Strategies*, tahap terakhir yang dilakukan yaitu proses seleksi dimana dalam proses ini dilakukan seleksi untuk masing – masing individu yang nanti akan dibawa untuk generasi yang berikutnya. Untuk metode seleksi yang diterapkan yaitu menggunakan metode *elitism*. Sebelum individu tersebut itu terpilih untuk dibawa ke generasi selanjutnya, individu tersebut harus diurutkan berdasarkan nilai *fitness* terbesar hingga terkecil yang kemudian akan diambil sebanyak populasi berdasarkan dari urutan tersebut.



Gambar 0.5 Fowchart Melakukan Proses Seleksi Populasi Baru

Sehingga langkah – langkah untuk dapat menghitung berapa nilai dari penalti gizi adalah sebagai berikut ini:

1. Gabungkan seluruh individu baik itu *parent* maupun *offspring* dengan disertakan nilai *fitness* dari tiap individu tersebut.
2. Sistem akan melakukan proses mengurutkan nilai *fitness* berdasarkan nilai *fitness* yang terbesar hingga terkecil.
3. Melakukan pemilihan individu dimana pemilihan ini dilakukan sesuai dengan jumlah populasi yang sudah ditentukan di awal agar individu yang terpilih tersebut akan menjadi *parent* pada generasi yang selanjutnya.
4. Proses seleksi selesai dengan hasil suatu populasi yang baru berdasarkan tahap pengurutan nilai *fitness* terbaik.

1.3 Perhitungan Manual

Pada perhitungan manual, dimisalkan data atlet adalah sebagai berikut:

| | |
|-------------------------------------|----------------|
| Jenis Kelamin | : Perempuan |
| Usia | : 20 |
| Berat Badan | : 49 |
| Berapa hari Latihan / Minggu | : 5 |
| Durasi Latihan / Hari (Menit) | : 120 |
| Aktifitas | : Aktif Ringan |
| Anggaran Harga yang Diinginkan (Rp) | : 50000 / hari |

Berdasarkan dari data tersebut maka hasil dari perhitungannya yaitu:

Langkah 1

$$\text{Basal Metabolic Rate (BMR)} = (14.7 * \text{BB}) + 679 = 1216.3 \text{ kkal}$$

$$\text{Specific Dynamic Action (SDA)} = 10\% * \text{BMR} = 121.63 \text{ kkal}$$

$$\text{BMR} + \text{SDA} = 1216.3 + 121.63 = 1337.93 \text{ kkal}$$

Langkah 2

Menentukan tingkat aktivitas fisik berdasarkan tabel (2.2) kemudian mengkalikan total BMR dan SDA dengan aktivitas fisik seperti pada persamaan (2.7). Diketahui bahwa atlet tergolong kedalam aktif ringan maka nilai dari aktivitas fisiknya yaitu 1.5.

$$(\text{BMR} + \text{SDA}) * \text{Aktivitas Fisik} = 1337.93 * 1.5 = 2006.895 \text{ kkal}$$

Langkah 3

Menghitung keluaran energi berdasarkan jenis olahraga. Diketahui bahwa atlet berlatih 5 hari seminggu dan setiap latihan berdurasi 120 menit kemudian berdasarkan berat badan maka diketahui kebutuhan kalori per menit yaitu 4. Sehingga didapatkan = $5 \text{ hari} * 120 \text{ menit} * 4 \text{ kkal} = 2400 \text{ kkal}$.

Selanjutnya jumlah tersebut dibagi dengan 7 hari agar dapat diketahui jumlah energi yang dibutuhkan dalam sehari berdasarkan durasi latihan = $2400 \text{ kkal} / 7 = 342.8571 \text{ kkal}$.

Kemudian jumlahkan hasil langkah 3 dengan langkah 2 = $342.8571 \text{ kkal} + 2006.895 \text{ kkal} = 2349.752 \text{ kkal}$.

Dengan demikian berdasarkan perhitungan dari langkah – langkah tersebut maka diketahui kebutuhan energi perhari pada atlet dalam contoh kasus diatas adalah sebanyak 2349.752 kkal .

Selanjutnya atlet dengan 2349.752 kkal membutuhkan jumlah zat gizi sebagai berikut:

Karbohidrat = $2349.752 * 0.6 = 1409.851 \text{ kkal} \approx 352.4628 \text{ gr}$

Protein = $2349.752 * 0.15 = 352.4628 \text{ kkal} \approx 88.11571 \text{ gr}$

Lemak = $2349.752 * 0.25 = 587.4380 \text{ kkal} \approx 65.27089 \text{ gr}$

1.3.1 Parameter Algoritma *Evolution Strategies*

Di dalam algoritma *Evolution Strategies* parameter – parameter yang diperlukan yaitu Populasi (μ), *Offspring* (λ), ukuran generasi serta priorias gizi.

Ukuran Populasi (μ) = 3

Ukuran *Offspring* (λ) = 1

Ukuran Generasi = 1

1.3.2 Generate Populasi Awal

Di dalam melakukan *generate* populasi awal akan dipermisalkan anggaran dana perhari adalah Rp. 50000,- serta bahan makanan yang dikonsumsi dalam 1 hari adalah sebagai berikut:

Tabel 0.1 Generate populasi awal berdasarkan inputan user

| Pilihan Menu Makanan | | Energi | Karbohidrat | Protein | Lemak | Harga |
|----------------------|-------------------|--------|-------------|---------|-------|-------|
| Pagi | Bihun | 381 | 9.3 | 0.3 | 0.1 | 1200 |
| | Jamur Kuping | 27 | 5 | 2.2 | 0.6 | 4500 |
| | Babat | 82.9 | 0 | 14.5 | 2.4 | 2600 |
| | Gado - gado | 137 | 21 | 6.1 | 3.2 | 2000 |
| | Apel | 59.1 | 15.3 | 0.2 | 0.4 | 1300 |
| Siang | Nasi Putih | 130 | 28.6 | 24 | 0.2 | 1150 |
| | Tahu | 115 | 2.5 | 9.7 | 8.5 | 2000 |
| | Ayam Goreng | 245 | 0.7 | 37.9 | 9 | 3250 |
| | Kangkung | 15 | 2.1 | 2.3 | 0.2 | 1250 |
| | Semangka | 32 | 7.2 | 0.6 | 0.4 | 700 |
| Malam | Kentang | 93 | 21.6 | 2 | 0.1 | 1350 |
| | Kacang Kapri | 83.8 | 15.6 | 5.4 | 0.2 | 1750 |
| | Telur Ayam | 155.1 | 1.1 | 12.6 | 10.6 | 1500 |
| | Buntil Daun Talas | 141 | 8 | 4.4 | 10.2 | 2800 |
| | Sawo | 82.9 | 21.4 | 0.3 | 0.6 | 1600 |

Pada saat melakukan proses *generate* populasi awal, proses tersebut merupakan proses dalam melakukan pembuatan populasi awal sejumlah ukuran populasi yang telah ditentukan. Kemudian jumlah gen dari masing – masing kromosom memiliki nilai yang sama sesuai pada jumlah bahan makanan yang dikonsumsi dalam 1 hari yaitu 15 yang terdiri dari 5 makanan untuk makan pagi, 5 makanan untuk makan siang dan 5 makanan untuk makan malam. Nilai dari gen – gen tersebut merupakan bilangan *random* yang dibangkitkan dengan *range* antara [0...1] dimana bilangan – bilangan tersebut akan merepresentasikan dari berat bahan makanan yang akan dikonsumsi.

Tabel 0.2 Random kromosom yang merepresentasikan berat berdasarkan dari populasi awal

| Parent | Kromosom | <i>fitness</i> |
|----------------|--|----------------|
| P ₁ | [0.6910 0.2720 0.1898 0.3963 0.7177 0.1908 0.0721 0.5692 0.3842 0.4402 0.2017 0.6015 0.6908 0.4852 0.4681] | 0.000543664 |
| P ₂ | [0.6377 0.6987 0.2983 0.8568 0.1951 0.149 0.1565 0.5113 0.1499 0.0551 0.5509 0.6572 0.4579 0.3061 0.3561] | 0.000252782 |
| P ₃ | [0.5373 0.8449 0.1704 0.8409 0.6473 0.1771 0.0085 0.1422 0.7562 0.1702 0.7047 0.0355 0.8659 0.7027 0.5506] | 0.000101972 |

1.3.3 Reproduksi

Pada saat proses dari inisialisasi awal telah selesai kemudian proses selanjutnya yaitu reproduksi. Akibat dari tipe siklus penelitian ini menggunakan tipe ES ($\mu + \lambda$), maka hanya terdapat proses mutasi tanpa rekombinasi pada saat reproduksi. Untuk setiap masing – masing gen yang terdapat pada inisialisasi awal akan mempunyai nilai level mutasi yang dilambangkan dengan (σ). Di mana

nilai dari level mutasi tersebut akan dibangkitkan secara acak dengan batasan antara [0...1], karena jumlah parent diketahui ada 3 maka jumlah child (*Offspring*) pun juga akan ada 3.

$P_1 = [0.6910 \ 0.2720 \ 0.1898 \ 0.3963 \ 0.7177 \ 0.1908 \ 0.0721 \ 0.5692 \ 0.3842 \ 0.4402 \ 0.2017 \ 0.6015 \ 0.6908 \ 0.4852 \ 0.4681]$

$\sigma_1 = [0.1946 \ 0.8814 \ 0.1367 \ 0.6349 \ 0.4047 \ 0.7653 \ 0.6426 \ 0.1487 \ 0.1944 \ 0.691 \ 0.3842 \ 0.7185 \ 0.8266 \ 0.557 \ 0.4012]$

Sehingga akan menghasilkan *Offspring* $C_1 = (x_1', x_2' \dots x_{15}')$ dan *sigma* mutasi $\sigma_1' = (\sigma_1', \sigma_2' \dots \sigma_{15}')$.

Berdasarkan persamaan (2.9) dan (2.10) maka diketahui:

$$X_1' = X_1 + \sigma_1 N(0,1)$$

$$X_2' = X_2 + \sigma_2 N(0,1)$$

Kemudian untuk nilai dari $N(0,1)$ akan diperoleh dengan menggunakan persamaan pada (2.11):

$$N(0,1) = \sqrt{-2 \cdot \ln r_1} \cdot \sin 2\pi r_2$$

Diketahui nilai $r_1 = 0.5031$ dan $r_2 = 0.1731$, maka

$$N(0,1) = \sqrt{-2 \cdot \ln 0.5031} \cdot (\sin(2\pi \cdot 0.1731)) = 1.037965127$$

Selanjutnya akan didapatkan nilai X_1' sebagai berikut:

$$X_1' = X_1 + \sigma_1 N(0,1) = 0.6910 + 0.1946 \cdot 1.037965127 = 0.892988014$$

Berikut adalah tabel individu hasil mutasi berdasarkan dari *parent* pada populasi awal.

Tabel 0.3 Individu hasil mutasi dari *parent*

| <i>Offspring</i> | Kromosom | <i>fitness</i> |
|------------------|---|----------------|
| C_1 | [0.8929 0.7653 0.2145 0.8632 0.6665 0.5655 0.9377 0.7152 0.5307 0.9667 0.1459 0.1330 1.1045 0.8433 0.3353 0.1751] | 3.44482E-05 |
| C_2 | [0.0063 0.6584 0.5884 0.5204 0.2925 0.1300 0.0881 0.5092 0.3769 0.0365 0.6021 0.5985 0.4284 0.1532 0.1934 0.8049] | 0.000378331 |
| C_3 | [0.8022 0.8775 0.1015 0.7706 0.2342 0.4884 0.0072 0.6921 0.9110 0.4615 0.1169 0.0586 0.4537 0.4661 0.2266 0.6661] | 7.11029E-05 |

Selanjutnya apabila nilai *fitness* dari *Offspring* lebih baik dari *parent* maka nilai σ akan dinaikkan dengan cara $\sigma' = \sigma \cdot 1.1$, namun apabila nilai *fitness* dari *Offspring* lebih buruk dari *fitness* pada *parent* maka nilai σ akan diturunkan dengan cara $\sigma' = \sigma \cdot 0.9$.

1.3.4 Perhitungan Penalti Gizi

Tabel 0.4 Penalti gizi pada gen individu pertama

| Gen ke- | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| P1 | 0.69 | 0.27 | 0.18 | 0.39 | 0.71 | 0.19 | 0.07 | 0.56 | 0.38 | 0.44 | 0.20 | 0.60 | 0.69 | 0.48 | 0.46 |

Pada penelitian ini, total penalti gizi terdiri berdasarkan jumlah dari penalti energi, penalti karbohidrat, penalti protein dan penalti lemak dengan memperhitungkan berat dari tiap – tiap gen yang akan dikalikan dengan 400 *gr* dengan menggunakan persamaan (2.13). Peneliti mengambil angka 400 sebagai acuan rata – rata berat bahan makanan perhari .

$$\text{berat} = \text{gen} \times 400 \text{ g}$$

Berdasarkan dari rumus tersebut maka berat sebenarnya pada makanan adalah:

$$\text{Gen 1: } 0.69 \times 400 = 276.4 \text{ gr}$$

$$\text{Gen 2: } 0.27 \times 400 = 108.8 \text{ gr}$$

Perhitungan dilanjutkan hingga gen ke 15.

Apabila perhitungan berat sebenarnya telah selesai dilakukan sampai gen ke 15, selanjutnya adalah memperhitungkan berat dari komponen penyusun energi, karbohidrat, protein dan lemak pada masing – masing makanan.

Misalnya pada bahan makanan bihun pada makan pagi yang seberat 276.4 *gr*, maka mengandung:

$$eg = \frac{\text{berat}}{100} \times \text{nilaiGiziEgi} = \frac{276.4}{100} \times 381 = 1053.08 \text{ Kal}$$

$$kb = \frac{\text{berat}}{100} \times \text{nilaiGiziKbi} = \frac{276.4}{100} \times 9.3 = 25.70 \text{ gr}$$

$$\text{prot} = \frac{\text{berat}}{100} \times \text{nilaiGiziProti} = \frac{276.4}{100} \times 0.3 = 0.82 \text{ gr}$$

$$\text{lem} = \frac{\text{berat}}{100} \times \text{nilaiGiziLemi} = \frac{276.4}{100} \times 0.1 = 0.27 \text{ gr}$$

Kemudian pada bahan makanan jamur kuping pada makan pagi yang seberat 108.8 *gr*, maka mengandung:

$$eg = \frac{\text{berat}}{100} \times \text{nilaiGiziEgi} = \frac{108.8}{100} \times 27 = 29.37 \text{ Kal}$$

$$kb = \frac{\text{berat}}{100} \times \text{nilaiGiziKbi} = \frac{108.8}{100} \times 5 = 5.44 \text{ gr}$$

$$\text{prot} = \frac{\text{berat}}{100} \times \text{nilaiGiziProti} = \frac{108.8}{100} \times 2.2 = 2.39 \text{ gr}$$

$$\text{lem} = \frac{\text{berat}}{100} \times \text{nilaiGiziLemi} = \frac{108.8}{100} \times 0.6 = 0.65 \text{ gr}$$

Apabila telah selesai dalam melakukan perhitungan berat tiap komponen penyusun pada masing – masing makanan maka dilanjutkan dengan menghitung total dari komponen yang telah diperhitungkan sebelumnya.

$$\text{Total Energi} = 1053.08 + 29.37 = 1082.45$$

$$\text{Total Karbohidrat} = 25.70 + 5.44 = 31.14$$

$$\text{Total Protein} = 0.82 + 2.39 = 3.21$$

$$\text{Total Lemak} = 0.27 + 0.6 = 0.87$$

Selanjutnya proses yang harus dilakukan adalah menghitung nilai dari penalti masing – masing jenis makanan dengan persamaan (2.18):

$$\text{penaltiMenuHarian} = |(egi - (kebEg * BPM))| + |(kbi - (kebKb * BPM))| + |(lemi - (kebLem * BPM))| + |(proti - (kebProt * BPM))|$$

Sehingga dari persamaan tersebut, akan didapatkan:

$$\text{Keb. Energi} = 2349.752 \text{ kkal}$$

$$\text{Karbohidrat} = 352.4628 \text{ gr}$$

$$\text{Lemak} = 65.27089 \text{ gr}$$

$$\text{Protein} = 88.11571 \text{ gr}$$

Maka selisih dari menu harian untuk makan pagi berdasarkan kandungan bahan makanan yang di rekomendasikan sistem dengan yang dibutuhkan yaitu:

$$\text{Energi} = |1532.23436 - (2349.752 * 0.25)| = 944.7963243$$

$$\text{Karbohidrat} = |108.35764 - (352.4628 * 0.25)| = 20.24193464$$

$$\text{Lemak} = |8.97224 - (65.27089 * 0.25)| = 2.446153661$$

$$\text{Protein} = |24.47508 - (88.11571 * 0.25)| = 7.345483214$$

$$\text{Total Penalty Menu Pagi} = 944.7963243 + 20.24193464 + 2.446153661 + 7.345483214 = 974.8298958$$

Ulangi langkah tersebut hingga mendapatkan total penalti menu siang dan menu malam agar dapat menghitung total penalti gizi. Setelah mendapatkan total penalti menu untuk tiap – tiap bahan pada *parent* 1, maka hitunglah total dari penalti tersebut untuk mendapatkan total penalti gizi dengan persamaan (2.19).

Tabel 0.5 Nilai penalti menu berdasarkan waktu makan

| Bahan | Nilai Penalti Menu |
|-------------|--------------------|
| Makan Pagi | 974.8298958 |
| Makan Siang | 220.4485441 |
| Makan Malam | 643.0926094 |

$$TotPenalti\ Gizi = penaltiMenuPagi_i + penaltiMenuSiang_i + penaltiMenuMalam_i$$

$$Total\ Penalti\ Gizi = 974.8298958 + 220.4485441 + 643.0926094 = 1838.371049$$

Dengan menggunakan langkah yang sama maka nilai total penalti gizi dari masing – masing individu yaitu:

Tabel 0.6 Nilai penalti gizi masing-masing individu

| Individu | Nilai Penalti Gizi |
|----------------|--------------------|
| P ₁ | 1838.371049 |
| P ₂ | 1828.080495 |
| P ₃ | 2521.624351 |
| C ₁ | 3660.595279 |
| C ₂ | 2642.186926 |
| C ₃ | 8387.139827 |

1.3.5 Perhitungan Penalti Harga

Di dalam melakukan perhitungan penalti harga maka persamaan yang digunakan yaitu persamaan (2.20):

$$totalHarga = \sum_{i=0}^n \frac{berat}{100} \times harga_i$$

Berdasarkan dari persamaan tersebut maka akan didapatkan harga bahan makanan sesuai dengan berat bahannya, seperti:

$$Bihun (276.4\ gr) = \frac{276.4}{100} \times 1200 = 3316.8$$

$$Jamur\ Keping (108.8\ gr) = \frac{108.8}{100} \times 4500 = 4896$$

Setelah menghitung harga dari masing – masing bahan makanan sampai pada gen terakhir maka selanjutnya adalah menghitung total harganya.

$$totalHarga = 3316.8 + 4896 + 1973.92 + 3170.4 + 3732.04 + 877.68 + 576.8 + 7399.6 + 1921 + 1232.56 + 1089.18 + 4210.5 + 4144.8 + 5434.24 + 2995.84 = 46971.36$$

Kemudian apabila total harga yang didapatkan lebih kecil dari anggaran yang diinginkan (pada kasus ini, data atlet menunjukkan bahwa atlet menganggarkan Rp. 50000,- / hari), maka nilai penalti harganya adalah 0.

Dengan melakukan cara yang sama, maka akan didapatkan hasil perhitungan penalti harga sebagai berikut:

Tabel 0.7 Hasil perhitungan penalti harga

| Individu | Nilai Penalty Harga |
|----------------|---------------------|
| P ₁ | 0 |
| P ₂ | 2126.9 |
| P ₃ | 7284 |
| C ₁ | 25367.51787 |
| C ₂ | 0 |
| C ₃ | 5675.98906 |

1.3.6 Menghitung Nilai *fitness*

Berdasarkan persamaan (2.12), maka rumus perhitungan *fitness* pada penelitian ini yaitu:

$$fitness = \frac{1}{penaltiGizi + penaltiHarga + 1}$$

Sehingga berdasarkan dari persamaan tersebut maka akan didapatkan:

$$fitness = \frac{1}{1838.371049 + 0 + 1} = 0.000543664$$

Lakukan cara yang sama untuk mendapatkan nilai *fitness* dari masing – masing kromosom yang lainnya, sehingga akan didapatkan seperti berikut ini:

Tabel 0.8 Nilai *fitness* dari tiap kromosom

| Individu | Nilai <i>fitness</i> |
|----------------|----------------------|
| P ₁ | 0.000543664 |
| P ₂ | 0.000252782 |
| P ₃ | 0.000101972 |
| C ₁ | 0.0000344482 |
| C ₂ | 0.000378331 |
| C ₃ | 0.000071103 |

1.3.7 Seleksi

Pada tahap terakhir yaitu seleksi, dilakukan untuk bisa mendapatkan individu baru sejumlah dari populasi awal yang nanti akan dijadikan sebagai *parent* dari generasi yang selanjutnya.

Tabel 0.9 Gabungan individu yang masuk ke tahap seleksi

| Parent | Kromosom | <i>fitness</i> |
|----------------------|---|----------------|
| P₁ | [0.6910 0.2720 0.1898 0.3963 0.7177 0.1908 0.0721 0.5692 0.3842 0.4402 0.2017 0.6015 0.6908 0.4852 0.4681] | 0.000543664 |
| P₂ | [0.6377 0.6987 0.2983 0.8568 0.1951 0.149 0.1565 0.5113 0.1499 0.0551 0.5509 0.6572 0.4579 0.3061 0.3561] | 0.000252782 |
| P₃ | [0.5373 0.8449 0.1704 0.8409 0.6473 0.1771 0.0085 0.1422 0.7562 0.1702 0.7047 0.0355 0.8659 0.7027 0.5506] | 0.000101972 |
| C₁ | [0.8929 0.7653 0.2145 0.8632 0.6665 0.5655 0.9377 0.7152 0.5307 0.9667 0.1459 0.1330 1.1045 0.8433 0.3353 0.1751] | 3.44482E-05 |
| C₂ | [0.0063 0.6584 0.5884 0.5204 0.2925 0.1300 0.0881 0.5092 0.3769 0.0365 0.6021 0.5985 0.4284 0.1532 0.1934 0.8049] | 0.000378331 |
| C₃ | [0.8022 0.8775 0.1015 0.7706 0.2342 0.4884 0.0072 0.6921 0.9110 0.4615 0.1169 0.0586 0.4537 0.4661 0.2266 0.6661] | 7.11029E-05 |

Apabila sistem telah mendapatkan nilai *fitness* dari masing – masing kromosom maka akan dilakukan sorting berdasarkan nilai terbesar ke terkecil (descending).

Tabel 0.10 Urutan individu berdasarkan dengan *fitness* yang terbesar

| Parent | Kromosom | <i>fitness</i> |
|----------------------|---|----------------|
| P₁ | [0.6910 0.2720 0.1898 0.3963 0.7177 0.1908 0.0721 0.5692 0.3842 0.4402 0.2017 0.6015 0.6908 0.4852 0.4681] | 0.000543664 |
| C₂ | [0.0063 0.6584 0.5884 0.5204 0.2925 0.1300 0.0881 0.5092 0.3769 0.0365 0.6021 0.5985 0.4284 0.1532 0.1934 0.8049] | 0.000378331 |
| P₂ | [0.6377 0.6987 0.2983 0.8568 0.1951 0.149 0.1565 0.5113 0.1499 0.0551 0.5509 0.6572 0.4579 0.3061 0.3561] | 0.000252782 |
| P₃ | [0.5373 0.8449 0.1704 0.8409 0.6473 0.1771 0.0085 0.1422 0.7562 0.1702 0.7047 0.0355 0.8659 0.7027 0.5506] | 0.000101972 |
| C₃ | [0.8022 0.8775 0.1015 0.7706 0.2342 0.4884 0.0072 0.6921 0.9110 0.4615 0.1169 0.0586 0.4537 0.4661 0.2266 0.6661] | 7.11029E-05 |
| C₁ | [0.8929 0.7653 0.2145 0.8632 0.6665 0.5655 0.9377 0.7152 0.5307 0.9667 0.1459 0.1330 1.1045 0.8433 0.3353 0.1751] | 3.44482E-05 |

Setelah mendapatkan urutan kromosom sesuai dengan nilai *fitness* yang terbesar ke terkecil maka akan dipilih individu yang mana individu tersebut akan menjadi *parent* pada generasi yang selanjutnya.

Tabel 0.11 Individu terpilih untuk dimasukan ke generasi selanjutnya

| Parent | Kromosom | fitness |
|--------|--|-------------|
| P_1 | [0.6910 0.2720 0.1898 0.3963 0.7177 0.1908 0.0721 0.5692 0.3842 0.4402 0.2017 0.6015 0.6908 0.4852 0.4681] | 0.000543664 |
| C_2 | [0.0063 0.6584 0.5884 0.5204 0.2925 0.1300 0.0881 0.5092 0.3769 0.0365 0.6021 0.5985 0.4284 0.1532 0.1934 0.8049] | 0.000378331 |
| P_2 | [0.6377 0.6987 0.2983 0.8568 0.1951 0.149 0.1565 0.5113 0.1499 0.0551 0.5509 0.6572 0.4579 0.3061 0.3561] | 0.000252782 |

1.4 Perancangan Uji Coba dan Evaluasi

Dalam mengevaluasi kerja sistem maka diperlukan uji coba agar bisa memperoleh nilai parameter yang optimal. Uji coba tersebut yaitu:

1. Uji coba dalam menentukan ukuran dari populasi (μ) serta *offspring* (λ) optimal.
2. Uji coba dalam menentukan berapa banyak generasi yang optimal.

1.4.1 Rancangan Pengujian Ukuran Populasi (μ) dan *offspring* (λ)

Di dalam pengujian ukuran μ dan λ ini akan dilakukan *input* pada *inputan* dari ukuran μ serta λ . Dalam hal ini akan dilakukan suatu pengujian untuk ukuran μ sebesar 5, 10, 15, 20, 25, 30 dengan ukuran λ sebesar 1μ , 2μ , 3μ , 4μ , 5μ , 6μ , 7μ , 8μ , 9μ dan 10μ . Tujuan dari pengujian ini yaitu agar dapat mengetahui pengaruh input dari ukuran μ serta ukuran λ . Apabila telah mendapatkan nilai *fitness* dari tiap – tiap kombinasi antara μ dan λ maka akan dapat terlihat rata – rata nilai *fitness* terbaik yang kemudian akan bisa dilakukan analisis dari pengaruh ukuran μ dan λ tersebut.

Tabel 0.12 Tabel rancangan uji coba banyaknya populasi

| Banyak Populasi (μ) | generasi , λ | | | | | Rata-rata nilai <i>fitness</i> |
|------------------------------|----------------------|---|---|-----|----|--------------------------------|
| | Percobaan ke- | | | | | |
| | 1 | 2 | 3 | ... | 10 | |
| 5 | | | | | | |
| 10 | | | | | | |
| 15 | | | | | | |
| 20 | | | | | | |
| 25 | | | | | | |
| 30 | | | | | | |

Tabel 0.13 Tabel rancangan uji coba ukuran *Offspring*

| Banyak Offspring (λ) | generasi , populasi | | | | | Rata-rata nilai <i>fitness</i> |
|--------------------------------|---------------------|---|---|-----|----|--------------------------------|
| | Percobaan ke- | | | | | |
| | 1 | 2 | 3 | ... | 10 | |
| 1 μ | | | | | | |
| 2 μ | | | | | | |
| 3 μ | | | | | | |
| 4 μ | | | | | | |
| 5 μ | | | | | | |
| ... | | | | | | |
| 10 μ | | | | | | |

1.4.2 Rancangan Pengujian Banyak Generasi

Dalam melakukan uji coba banyaknya generasi maka ukuran dari generasi yang akan digunakan yaitu 15, 20, 25, 30, 35, 40, 45, 50, 55, 60 dimana dari tiap – tiap ukuran generasi tersebut akan dilakukan pengujian sebanyak 10 kali yang selanjutnya akan diambil nilai rata – ratanya.

Tabel 0.14 Tabel rancangan uji coba ukuran generasi

| Banyak Generasi | μ, λ | | | | | Rata- rata nilai <i>fitness</i> |
|--------------------|----------------|---|---|-----|----|--|
| | Percobaan ke- | | | | | |
| | 1 | 2 | 3 | ... | 10 | |
| 15 | | | | | | |
| 20 | | | | | | |
| 25 | | | | | | |
| 30 | | | | | | |
| 35 | | | | | | |
| 40 | | | | | | |
| 45 | | | | | | |
| 50 | | | | | | |
| 55 | | | | | | |
| 60 | | | | | | |

IMPLEMENTASI

Di dalam bab ini akan membahas mengenai pengimplementasian dari sistem optimasi komposisi bahan makanan untuk atlet menembak dengan menggunakan algoritma *Evolution Strategies* dengan tipe siklus ($\mu+\lambda$).

1.1 Lingkungan Implementasi

Lingkungan implementasi ini akan membahas tentang lingkungan implementasi perangkat keras serta perangkat lunak yang digunakan pada saat pembuatan sistem optimasi komposisi bahan makanan untuk atlet menembak dengan menggunakan algoritma *Evolution Strategies*.

1.1.1 Lingkungan Perangkat Keras

Pada saat pembuatan sistem ini, perangkat keras yang digunakan adalah sebagai berikut:

1. Processor Intel(R) Core(TM) i7-7500U CPU 3.50GHz
2. RAM 8,00 GB
3. Monitor 14"

1.1.2 Lingkungan Perangkat Lunak

Pada saat pembuatan sistem ini, perangkat lunak yang digunakan adalah sebagai berikut:

1. Sistem operasi *Windows 10* 64-bit
2. Netbeans IDE 7.3.1

1.2 Implementasi Sistem

Di dalam sub bab ini akan dibahas mengenai implementasi kode program yang diterapkan dalam sistem optimasi komposisi bahan makanan untuk atlet menembak. Kode program yang dibahas meliputi proses *generate* populasi awal, proses mutasi, proses menghitung *fitness* serta proses seleksi.

1.2.1 Pengambilan Data Makanan

```
1 public class Makanan {
2     private String[] nama = new String[25];
3     private double[][] dataSet = new double[25][5];
4
5     public String[] getNama() {
6         try {
7             InputStream in =
8 this.getClass().getResourceAsStream("makanan1.txt");
9             BufferedReader br = new BufferedReader(new
10 InputStreamReader(in));
11             String line;
12             int j = 0;
13             while ((line = br.readLine()) != null) {
14                 nama[j] = line;
```

```

15         j++;
16     }
17     br.close();
18     } catch (Exception e) {
19         System.out.println("Error: " + e.getMessage());
20     }
21     return nama;
22 }
23
24 public double[][] getDataSet() {
25     try {
26         InputStream in =
27 this.getClass().getResourceAsStream("Makanan.txt");
28         BufferedReader br = new BufferedReader(new
29 InputStreamReader(in));
30         String line;
31         int j = 0;
32         while ((line = br.readLine()) != null) {
33             int start = line.indexOf(",");
34             String sub = line.substring(start);
35             String[] vertex = sub.split(" ");
36
37             dataSet[j][0] = 0 + Double.parseDouble(vertex[0]);
38             dataSet[j][1] = 0 + Double.parseDouble(vertex[1]);
39             dataSet[j][2] = 0 + Double.parseDouble(vertex[2]);
40             dataSet[j][3] = 0 + Double.parseDouble(vertex[3]);
41             dataSet[j][4] = 0 + Double.parseDouble(vertex[4]);
42             j++;
43         }
44         br.close();
45     } catch (Exception e) {
46         System.out.println("Error: " + e.getMessage());
47     }
48     return dataSet;
49 }

```

Gambar 0.1 Source Code Pengambilan Data Makanan

Penjelasan *source code* dari Gambar 5.1:

- Baris 2 – 4 adalah proses pendeklarasian variable nama dan dataset yang bertipe array.
- Baris 9 – 10 adalah fungsi yang digunakan untuk dapat membaca karakter dari lokasi file yang diinginkan supaya bisa di-*input*-kan ke dalam sistem.
- Baris 5 – 22 adalah proses pengambilan data nama makanan yang berasal dari lokasi file dengan tipe (.txt).
- Baris 28 – 29 adalah fungsi yang digunakan untuk dapat membaca bilangan dari lokasi file yang diinginkan supaya bisa di-*input*-kan ke dalam sistem.
- Baris 24 – 49 adalah proses pengambilan dataset makanan yang berasal dari lokasi file dengan tipe (.txt) yang mana file tersebut berisikan tentang kandungan gizi dan harga.

1.2.2 Implementasi *Generate* Populasi Awal

```

1 public class ES {
2     int populasi = individu;
3     int kromosomlength = 45;
4     double[][] genes = new double[populasi][kromosomlength];
5     double[][] sigma = new double[populasi][kromosomlength];
6     double[][] konversigen = new double[populasi][kromosomlength];
7     for (int i = 0; i < populasi; i++) {
8         for (int j = 0; j < kromosomlength; j++) {

```

```

9         int nilaiTerendah = 0, nilaiTertinggi = 1000;
10        genes[i][j] = random.nextInt(nilaiTertinggi - nilaiTerendah +
11        1) + nilaiTerendah;
12    }
13    }
14    for (int i = 0; i < populasi; i++) {
15        for (int j = 0; j < kromosomlength; j++) {
16            int nilaiTerendah = 0, nilaiTertinggi = 1000;
17            sigma[i][j] = random.nextInt(nilaiTertinggi - nilaiTerendah +
18            1) + nilaiTerendah;
19        }
20    }
21    konversigen = a.konversi(genes);
22
23    sigmakonversi = a.konversi(sigma);
24
25    public double[][] konversi(double[][] genes) {
26        int panjangkromosom = genes[0].length;
27        int popsize = genes.length;
28        double hasil[][] = new double[popsize][panjangkromosom];
29        for (int i = 0; i < popsize; i++) {
30            for (int j = 0; j < panjangkromosom; j++) {
31                hasil[i][j] = (genes[i][j] / 1000 * (1));
32            }
33        }

```

Gambar 0.2 Source Code Generate Populasi Awal

Penjelasan *source code* dari Gambar 5.2:

- Baris 2 – 6 adalah proses pendeklarasian variable populasi, kromosomlength serta genes, *sigma* dan konversigen yang memiliki tipe array
- Baris 7 – 13 adalah proses random kromosom awal yang dibangkitkan dengan range 0 – 1000 dengan tujuan mendapatkan 3 angka dibelakang koma dengan panjang individu kromosom sesuai jumlah bahan makanan yang dimasukan.
- Baris 14 – 20 adalah proses random *sigma* awal yang dibangkitkan dengan range 0 – 1000 dengan tujuan mendapatkan 3 angka dibelakang koma dengan panjang sesuai kromosom.
- Baris 21 - 33 adalah proses konversi nilai gen dan *sigma* pada saat pembangkitan nilai random agar mendapatkan nilai dengan range 0 – 1.

1.2.3 Implementasi Proses Mutasi

```

1    public class ES {
2        double[][] r1 = new double[lamda][kromosomlength];
3        double[][] r2 = new double[lamda][kromosomlength];
4        double[][] N = new double[lamda][kromosomlength];
5        double[][] C = new double[lamda][kromosomlength];
6        double[][] konversigen = new double[populasi][kromosomlength];
7        double[][] sigmakonversi = new double[populasi][kromosomlength];
8        double[][] r1konversi = new double[lamda][kromosomlength];
9        double[][] r2konversi = new double[lamda][kromosomlength];
10       double[][] populasigabungan = new double[populasi +
11       lamda][kromosomlength];
12       double[][] sigmaadaption = new double[populasi][kromosomlength];
13       double[] fitness = new double[populasi];
14       double[] fitnessmutasi = new double[lamda];
15
16       for (int i = 0; i < lamda; i++) {
17           for (int j = 0; j < kromosomlength; j++) {
18               int nilaiTerendah = 0, nilaiTertinggi = 1000;

```

```

19         r1[i][j] = random.nextInt(nilaiTertinggi - nilaiTerendah
20 + 1) + nilaiTerendah;
21     }
22 }
23     for (int i = 0; i < lamda; i++) {
24         for (int j = 0; j < kromosomlength; j++) {
25             int nilaiTerendah = 0, nilaiTertinggi = 1000;
26             r2[i][j] = random.nextInt(nilaiTertinggi - nilaiTerendah
27 + 1) + nilaiTerendah;
28         }
29     }
30     r1konversi = a.konversi(r1);
31     r2konversi = a.konversi(r2);
32     N = a.NilaiN(r1konversi, r2konversi);
33     C = a.mutasi(konversigen, N, sigmakonversi, offspring, populasi);
34
35
36     sigmaadaption = a.selfAdaption(fitness, fitnessmutasi,
37 sigmakonversi, offspring);
38     for (int i = 0; i < populasi; i++) {
39         System.out.print("\tS" + (i + 1) + "\t\t| ");
40         for (int j = 0; j < kromosomlength; j++) {
41             System.out.print(sigmaadaption[i][j] + " ");
42         }
43         System.out.println("");
44     }
45
46     sigmakonversi = sigmaadaption;
47
48
49     public double[][] konversi(double[][] genes) {
50         int panjangkromosom = genes[0].length;
51         int popsize = genes.length;
52         double hasil[][] = new double[popsize][panjangkromosom];
53         for (int i = 0; i < popsize; i++) {
54             for (int j = 0; j < panjangkromosom; j++) {
55                 hasil[i][j] = (genes[i][j] / 1000 * (1));
56             }
57         }
58         return hasil;
59     }
60
61     public double[][] NilaiN(double[][] r1, double[][] r2) {
62         int panjangkromosom = r1[0].length;
63         int popsize = r1.length;
64         double hasil[][] = new double[popsize][panjangkromosom];
65         for (int i = 0; i < popsize; i++) {
66             for (int j = 0; j < panjangkromosom; j++) {
67                 hasil[i][j] = Math.sqrt(-2 * Math.log(r1[i][j]) /
68 Math.log(Math.E))
69                     * Math.sin(2 * Math.PI * r2[i][j]);
70             }
71         }
72         return hasil;
73     }
74
75     public double[][] mutasi(double[][] populasi, double[][] N, double[][]
76 sigma, int offspring, int individu) {
77         int panjangkromosom = populasi[0].length;
78         int popsize = populasi.length;
79         int lamda = offspring * individu;
80         double hasil[][] = new double[lamda][panjangkromosom];
81         int indexpop = 0;
82         int temp = 0;
83         int anak = offspring;
84
85         for (int i = 0; i < lamda; i++) {
86             if (i < anak) {
87                 indexpop = temp;
88             } else {
89                 temp = indexpop + 1;
90                 indexpop = temp;
91                 anak = anak + offspring;
92             }

```

```

93         for (int j = 0; j < panjangkromosom; j++) {
94             hasil[i][j] = Math.abs(populasi[indexpop][j] + (N[i][j] *
95             sigma[indexpop][j]));
96         }
97     }
98     return hasil;
99 }
100
101 public double[][] gabungIndividu(double[][] populasi, double[][] mutasi)
102 {
103     int panjangkromosom = populasi[0].length;
104     int popsize = populasi.length;
105     int mut = mutasi.length;
106     double hasil[][] = new double[popsize + mut][panjangkromosom];
107     for (int i = 0; i < popsize; i++) {
108         for (int j = 0; j < panjangkromosom; j++) {
109             hasil[i][j] = populasi[i][j];
110         }
111     }
112     for (int i = 0; i < mut; i++) {
113         for (int j = 0; j < panjangkromosom; j++) {
114             hasil[i + popsize][j] = mutasi[i][j];
115         }
116     }
117     return hasil;
118 }
119
120 public double[] gabungFitness(double[] fitness, double[] fitnessmutasi) {
121     int popsize = fitness.length;
122     int mut = fitnessmutasi.length;
123     double hasil[] = new double[popsize + mut];
124     for (int i = 0; i < popsize; i++) {
125         hasil[i] = fitness[i];
126     }
127     for (int i = 0; i < mut; i++) {
128         hasil[i + popsize] = fitnessmutasi[i];
129     }
130     return hasil;
131 }
132
133 public double[][] selfAdaption(double[] fitness, double[] fitnessmutasi,
134 double[][] sigma, int offspring) {
135     int popsize = fitnessmutasi.length;
136     int panjangkromosom = sigma[0].length;
137     int sig = fitness.length;
138     double hasil[][] = new double[sig][panjangkromosom];
139     int indexflag[] = new int[sig];
140     int indexpop = 0;
141     int temp = 0;
142     int anak = offspring;
143     double sort = fitness[0];
144
145     for (int i = 0; i < popsize; i++) {
146         if (i < anak) {
147             indexpop = temp;
148             indexflag[indexpop] = 0;
149             if (fitnessmutasi[i] > sort) {
150                 indexflag[indexpop] = 1;
151             } else {
152                 if (indexflag[indexpop] == 0) {
153                     indexflag[indexpop] = 0;
154                 } else {
155                     indexflag[indexpop] = 1;
156                 }
157             }
158         } else {
159             temp = indexpop + 1;
160             indexpop = temp;
161             anak = anak + offspring;
162             sort = fitness[indexpop];
163         }
164         if (indexflag[indexpop] == 1) {
165             for (int j = 0; j < panjangkromosom; j++) {

```

| | |
|-----|--|
| 166 | hasil[indexpop][j] = sigma[indexpop][j] * 1.1; |
| 167 | } |
| 168 | } else { |
| 169 | for (int j = 0; j < panjangkromosom; j++) { |
| 170 | hasil[indexpop][j] = sigma[indexpop][j] * 0.9; |
| 171 | } |
| 172 | } |
| 173 | } |
| 174 | return hasil; |
| 175 | } |

Gambar 0.3 Source Code Proses Mutasi

Penjelasan *source code* dari Gambar 5.3:

- Baris 2 – 14 adalah proses pendeklarasian variable yang digunakan dengan tipe data array.
- Baris 16 – 29 adalah proses random r1 dan r2 awal yang dibangkitkan dengan range 0 – 1000 dengan tujuan mendapatkan 3 angka dibelakang koma dengan panjang sesuai kromosom.
- Baris 30 – 31 adalah proses memanggil method konversi nilai r1 dan r2.
- Baris 32 adalah proses menghitung nilai N dengan memanggil method nilaiN.
- Baris 33 adalah proses menghitung mutasi dengan memanggil method mutasi.
- Baris 36 – 44 adalah proses memanggil method *self-adaption* untuk merubah nilai *sigma* berdasarkan dari perbandingan nilai *fitness* induk dan anaknya
- Baris 46 adalah method memasukan nilai *sigma* baru untuk dihitung pada proses berikutnya.
- Baris 49 – 59 melakukan konversi nilai pada saat pembangkitan nilai *random* agar mendapatkan nilai dengan range 0 – 1
- Baris 61 – 73 menghitung nilai N berdasarkan dari pembangkitan nilai r1 dan r2 secara random untuk digunakan dalam proses mutasi
- Baris 75 – 99 proses menghitung nilai gen kromosom pada masing – masing anak yang dihasilkan pada saat proses mutasi
- Baris 101 – 131 adalah proses penggabungan individu dan nilai *fitness* dari populasi awal dengan individu dan *fitness* yang didapatkan dari hasil mutasi
- Baris 133 – 175 adalah proses menghitung nilai *self-adaption* pada *sigma* berdasarkan dari perbandingan nilai *fitness parent* dengan *offspring*.

1.2.4 Implementasi Menghitung Nilai *fitness*

1.2.4.1 Implementasi Menghitung Kandungan Gizi Sebenarnya

```

1 public class Makanan {
2     private String[] nama = new String[25];
3     private double[][] dataSet = new double[25][5];
4     public double[][] dataHasil = new double[9][5];
5     private int index[] = new int[9];
6     double[] hasil = new double[9];
7     public double [] berat(double gen, double gen5, double gen10, double
8     gen15, double gen20, double gen25, double gen30, double gen35, double gen40)
9     {
10         hasil[0] = gen * 400;
11         hasil[1] = gen5 * 400;
12         hasil[2] = gen10 * 400;
13         hasil[3] = gen15 * 400;
14         hasil[4] = gen20 * 400;
15         hasil[5] = gen25 * 400;
16         hasil[6] = gen30 * 400;
17         hasil[7] = gen35 * 400;
18         hasil[8] = gen40 * 400;
19         return hasil;
20     }
21
22     public double[][] gizi() {
23         for (int i = 0; i < 9; i++) {
24             dataHasil[i][0] = (hasil[i] / 100) * dataSet[index[i]][0];
25         }
26         for (int i = 0; i < 9; i++) {
27             dataHasil[i][1] = (hasil[i] / 100) * dataSet[index[i]][1];
28         }
29         for (int i = 0; i < 9; i++) {
30             dataHasil[i][2] = (hasil[i] / 100) * dataSet[index[i]][2];
31         }
32         for (int i = 0; i < 9; i++) {
33             dataHasil[i][3] = (hasil[i] / 100) * dataSet[index[i]][3];
34         }
35         for (int i = 0; i < 9; i++) {
36             dataHasil[i][4] = (hasil[i] / 100) * dataSet[index[i]][4];
37         }
38         return dataHasil;
39     }

```

Gambar 0.4 Source Code Menghitung Kandungan Gizi Sebenarnya

Penjelasan *source code* dari Gambar 5.4:

- Baris 2 – 6 adalah pendeklarasian variable pada *class* Makanan.
- Baris 7 – 20 adalah proses menghitung berat dari tiap gen untuk mengetahui berat makanan yang sebenarnya.
- Baris 22 – 39 adalah proses menghitung berat dari komponen penyusun energi, karbohidrat, protein dan lemak pada masing – masing makanan.

1.2.4.2 Implementasi Menghitung Penalti Gizi

```

1 public class Es_Gizi {
2     double[][] genes = new double[45][5];
3     double[][] makanan = new double[9][5];
4     double pinalti = 0;
5     double harga = 0;
6     double penergi = 0;
7     double pkarbohidrat = 0;
8     double pprotein = 0;
9     double plemak = 0;
10    double pharga = 0;
11    Makanan m = new Makanan();
12    Gizi g = new Gizi();

```

```

13
14     public double jumlahPinalti(double[][] hasil, double energi, double
15     karbohidrat, double protein, double lemak) {
16         for (int i = 0; i < 3; i++) {
17             penergi = Math.abs(hasil[i][0] - (energi * 0.25)) + penergi;
18             pkarbohidrat = Math.abs(hasil[i][1] - (karbohidrat * 0.25)) +
19             pkarbohidrat;
20             pprotein = Math.abs(hasil[i][2] - (protein * 0.25)) + pprotein;
21             plemak = Math.abs(hasil[i][3] - (lemak * 0.25)) + plemak;
22         }
23
24         double energiS = 0, karbohidratS = 0, proteinS = 0, lemakS = 0;
25
26         for (int i = 3; i < 6; i++){
27             energiS = Math.abs(hasil[i][0] - (energi * 0.3)) + energiS;
28             karbohidratS = Math.abs(hasil[i][1] - (karbohidrat * 0.3)) +
29             karbohidratS;
30             proteinS = Math.abs(hasil[i][2] - (protein * 0.3)) + proteinS;
31             lemakS = Math.abs(hasil[i][3] - (lemak * 0.3)) + lemakS;
32         }
33
34         double energiM = 0, karbohidratM = 0, proteinM = 0, lemakM = 0;
35
36         for (int i = 0; i < 3; i++) {
37             energiM = Math.abs(hasil[i+6][0] - (energi * 0.25)) + energiM;
38             karbohidratM = Math.abs(hasil[i+6][1] - (karbohidrat * 0.25)) +
39             karbohidratM;
40             proteinM = Math.abs(hasil[i+6][2] - (protein * 0.25)) + proteinM;
41             lemakM = Math.abs(hasil[i+6][3] - (lemak * 0.25)) + lemakM;
42         }
43
44         pinalti = penergi + pkarbohidrat + pprotein + plemak + energiS +
45         karbohidratS + proteinS + lemakS + energiM + karbohidratM + proteinM +
46         lemakM;
47         return pinalti;
48     }

```

Gambar 0.5 Source Code Menghitung Penalti Gizi

Penjelasan *source code* dari Gambar 5.5:

- Baris 2 – 12 adalah pendeklarasian variable pada *class* ES_Gizi.
- Baris 16 – 42 adalah proses menghitung nilai selisih dari menu harian tiap jendela makan dengan berdasarkan kandungan bahan makanan dan nilai Bobot Prioritas Menu (BPM) nya.
- Baris 44 – 48 adalah proses menghitung penalti gizi dengan menjumlahkan nilai selisih tiap menu.

1.2.4.3 Implementasi Menghitung Penalti Harga dan Nilai *fitness*

```

1 public class ES {
2     public double hitungFitness() {
3         Es_Gizi e = new Es_Gizi();
4         double pinaltiGizi = 0;
5         double Harga = 0;
6         double pinaltiHarga = 0;
7         double fitness = 0;
8
9         pinaltiGizi = e.hasilPinaltiGizi(genes[0], genes[1], genes[2],
10        genes[3], genes[4], genes[5], genes[6], genes[7], genes[8], genes[9],
11        genes[10], genes[11], genes[12], genes[13], genes[14], genes[15],
12        genes[16], genes[17], genes[18], genes[19], genes[20],
13        genes[21], genes[22], genes[23], genes[24], genes[25], genes[26], genes[27],
14        genes[28], genes[29], genes[30],
15        genes[31], genes[32], genes[33], genes[34], genes[35],
16        genes[36], genes[37], genes[38], genes[39], genes[40], genes[41], genes[42],
17        genes[43], genes[44],
18        nama, nama1, nama2, nama3, nama4, nama5, nama6, nama7, nama8,
19        nama9, nama10, nama11, nama12, nama13,

```

```

20         nama14, nama15, nama16, nama17, nama18, nama19, nama20,
21     nama21, nama22, nama23, nama24, nama25, nama26, nama27, nama28, nama29,
22         nama30, nama31, nama32, nama33, nama34, nama35, nama36,
23     nama37, nama38, nama39, nama40, nama41, nama42, nama43, nama44, JK, umur, BB,
24         latihan, durasi, aktif, anggaran);
25
26     Harga = e.hasilPinaltiHarga();
27
28     if (Harga > anggaran) {
29         pinaltiHarga = Harga - anggaran;
30     } else {
31         pinaltiHarga = 0;
32     }
33
34     fitness = 1 / (pinaltiHarga + pinaltiGizi + 1);
35     return fitness;
36 }

```

Gambar 0.6 Implementasi Menghitung Penalti Harga dan Nilai *fitness*

Penjelasan *source code* dari Gambar 5.6:

- Baris 3 – 7 adalah pendeklarasian variable untuk hitung nilai *fitness*.
- Baris 9 – 24 adalah method untuk memanggil hasil penalti gizi yang berada pada *class ES_Gizi*.
- Baris 26 – 32 adalah proses untuk menghitung nilai penalti harga jika harga yang di rekomendasikan lebih besar daripada yang dianggarkan maka sistem akan menghitung nilai selisihnya namun jika tidak maka penalti harga = 0
- Baris 34 – 46 adalah rumus perhitungan nilai *fitness*.

1.2.5 Implementasi Proses Seleksi

```

1     public class ES {
2
3
4         public double[][] gabungIndividu(double[][] populasi, double[][] mutasi)
5     {
6         int panjangkromosom = populasi[0].length;
7         int popsize = populasi.length;
8         int mut = mutasi.length;
9         double hasil[][] = new double[popsize + mut][panjangkromosom];
10        for (int i = 0; i < popsize; i++) {
11            for (int j = 0; j < panjangkromosom; j++) {
12                hasil[i][j] = populasi[i][j];
13            }
14        }
15        for (int i = 0; i < mut; i++) {
16            for (int j = 0; j < panjangkromosom; j++) {
17                hasil[i + popsize][j] = mutasi[i][j];
18            }
19        }
20
21        return hasil;
22    }
23
24    public double[] gabungFitness(double[] fitness, double[] fitnessmutasi) {
25        int popsize = fitness.length;
26        int mut = fitnessmutasi.length;
27        double hasil[] = new double[popsize + mut];
28        for (int i = 0; i < popsize; i++) {
29            hasil[i] = fitness[i];
30        }
31        for (int i = 0; i < mut; i++) {
32            hasil[i + popsize] = fitnessmutasi[i];
33        }

```

```

34         return hasil;
35     }
36
37     public double[] gabungHarga(double[] harga induk, double[] harga anak) {
38         int popsize = harga induk.length;
39         int mut = harga anak.length;
40         double hasil[] = new double[popsize + mut];
41         for (int i = 0; i < popsize; i++) {
42             hasil[i] = harga induk[i];
43         }
44         for (int i = 0; i < mut; i++) {
45             hasil[i + popsize] = harga anak[i];
46         }
47         return hasil;
48     }
49
50     public double[][] ElitismSelection(double[][] populasigabungan, int
51     Pop_Size, double[] fitnessgabung) {
52         int panjangkromosom = populasigabungan[0].length;
53         int popsize = populasigabungan.length;
54         double[][] Hasil = new double[Pop_Size][panjangkromosom + 1];
55         int[] hasilFitness = new int[Pop_Size];
56
57         int[] indeksPertama = new int[popsize];
58         for (int i = 0; i < popsize; i++) {
59             indeksPertama[i] = i;
60         }
61
62         for (int i = 0; i < popsize; i++) {
63             double TempFitness = fitnessgabung[i];
64             int TempindeksPertama = indeksPertama[i];
65             double[][] TempHasil = new double[1][panjangkromosom];
66
67             for (int j = (i + 1); j < popsize; j++) {
68                 if (fitnessgabung[j] > TempFitness) {
69                     TempFitness = fitnessgabung[j];
70                     fitnessgabung[j] = fitnessgabung[i];
71                     fitnessgabung[i] = TempFitness;
72                     TempindeksPertama = indeksPertama[j];
73                     indeksPertama[j] = indeksPertama[i];
74                     indeksPertama[i] = TempindeksPertama;
75                     for (int k = 0; k < panjangkromosom; k++) {
76                         TempHasil[0][k] = populasigabungan[j][k];
77                     }
78                     for (int k = 0; k < panjangkromosom; k++) {
79                         populasigabungan[j][k] = populasigabungan[i][k];
80                     }
81                     for (int k = 0; k < panjangkromosom; k++) {
82                         populasigabungan[i][k] = TempHasil[0][k];
83                     }
84                 }
85             }
86         }
87
88         for (int i = 0; i < Pop_Size; i++) {
89             for (int j = 0; j < panjangkromosom; j++) {
90                 Hasil[i][j] = populasigabungan[i][j];
91             }
92         }
93         for (int i = 0; i < Pop_Size; i++) {
94             Hasil[i][panjangkromosom] = fitnessgabung[i];
95         }

```

Gambar 0.7 Implementasi Proses Seleksi

Penjelasan *source code* dari Gambar 5.7:

- Baris 4 – 22 adalah method untuk menggabungkan individu dari populasi awal dengan yang didapat dari hasil mutasi.
- Baris 24 – 35 adalah method untuk menggabungkan nilai *fitness* dari populasi awal dengan yang didapat dari hasil mutasi.

- Baris 37 – 48 adalah method untuk menggabungkan total harga dari populasi awal dengan yang didapat dari hasil mutasi.
- Baris 50 – 60 adalah proses seleksi dengan menggunakan metode *elitism*
- Baris 62 – 86 adalah proses sorting berdasarkan nilai *fitness* yang terbesar.
- Baris 88 – 95 adalah proses pengambilan kromosom tersorting berdasarkan dari nilai *fitness* nya sebanyak ukuran populasi awal untuk digunakan pada generasi selanjutnya.



ANALISA DAN PENGUJIAN

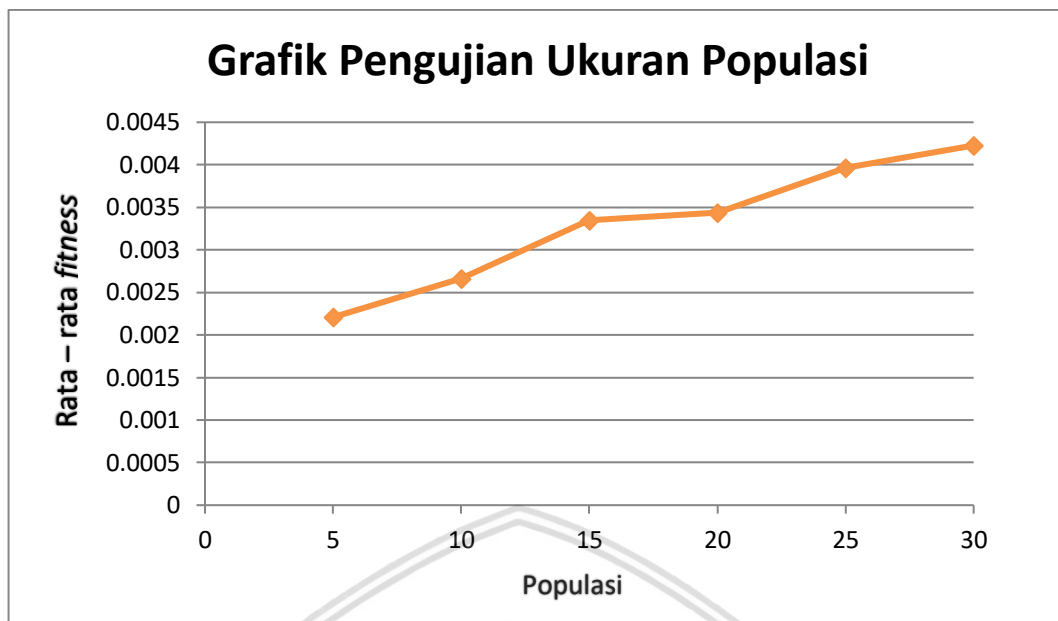
Pada bab ini berisi mengenai hasil dari pengujian dan analisa sistem optimasi komposisi bahan makanan untuk atlet menembak dengan menggunakan algoritma *Evolution Strategies*. Pengujian pertama dilakukan berdasarkan ukuran populasi (*miu*) dengan tujuan mengetahui seberapa pengaruh ukuran populasi (*miu*), pengujian kedua yaitu berdasarkan ukuran *offspring* (*lambda*) dengan tujuan agar dapat mengetahui pengaruh dari ukuran *offspring* (*lambda*), pengujian ketiga dilakukan berdasarkan banyaknya generasi dengan tujuan untuk dapat mengetahui seberapa pengaruh masukan banyaknya generasi terhadap *fitness* yang dihasilkan dengan menggunakan ukuran populasi dan *offspring* terbaik yang telah didapatkan dari pengujian sebelumnya, pengujian terakhir berdasarkan dari pengujian parameter terbaik yang telah didapatkan dari pengujian sebelumnya dengan tujuan untuk dapat mengetahui seberapa pengaruh parameter tersebut guna mendapatkan solusi yang optimal.

1.1 Hasil dan Analisa Pengujian Ukuran Populasi (μ) dan *offspring* (λ)

Di dalam pengujian ukuran populasi (μ) dan *offspring* (λ) ini akan dilakukan pengujian sebanyak 10 kali yang kemudian akan diambil nilai rata – rata *fitness* nya. Nilai *fitness* dari variasi masukan ukuran populasi (*miu*) dan ukuran *offspring* (*lambda*) dapat dilihat pada Tabel 6.1 dan Tabel 6.2.

Tabel 0.1 Pengujian Ukuran Populasi

| Banyak Populasi | generasi = 50 ; lambda = 5μ | | | | | | | | | | rata-rata |
|-----------------|-----------------------------|------|------|------|------|------|------|------|------|------|-----------|
| | percobaan ke- | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 5 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,002 |
| | 2152 | 1396 | 3112 | 1805 | 1705 | 2932 | 2032 | 2339 | 2393 | 2224 | 209 |
| 10 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,002 |
| | 3583 | 2426 | 3119 | 2928 | 2533 | 2408 | 2837 | 2332 | 2256 | 219 | 661 |
| 15 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,003 |
| | 3825 | 2733 | 3287 | 3027 | 3469 | 3101 | 3428 | 4009 | 3453 | 3113 | 345 |
| 20 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,003 |
| | 3487 | 3166 | 3598 | 41 | 344 | 3571 | 375 | 2786 | 3272 | 3187 | 436 |
| 25 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,003 |
| | 3127 | 4135 | 3207 | 4229 | 4248 | 4364 | 4214 | 4737 | 3336 | 4057 | 965 |
| 30 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,004 |
| | 4081 | 5004 | 4696 | 4581 | 4165 | 473 | 388 | 4056 | 3715 | 3375 | 228 |

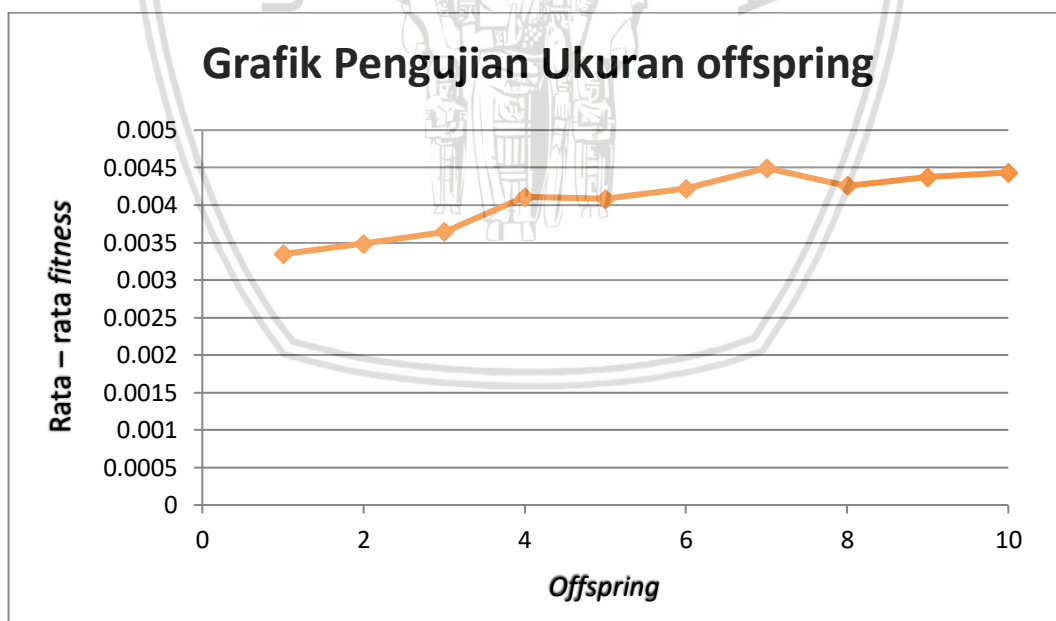


Gambar 0.1 Grafik Pengujian Ukuran Populasi

Di dalam pengujian populasi ini, dilakukan dengan memberikan variasi masukan jumlah populasi sebanyak 5, 10, 15, 20, 25 dan 30 dengan generasi sebanyak 50 dan ukuran *offspring* adalah 5 μ dengan masing – masing diujikan sebanyak 10 kali yang kemudian diambil nilai rata – rata *fitness* nya dimana nantinya akan terlihat inputan populasi terbaik berdasarkan nilai rata – rata *fitness* yang terbesar. Berdasarkan hasil dari Tabel 6.1 dan Gambar 6.1, terlihat bahwa ukuran populasi mempengaruhi hasil nilai *fitness* solusi yang dihasilkan. Sehingga hal tersebut menunjukkan bahwa seiring dengan bertambahnya ukuran populasi maka nilai *fitness* yang dihasilkannya pun juga akan semakin besar karena dengan semakin banyaknya populasi maka ruang lingkup pencariannya juga akan semakin luas.

Tabel 0.2 Pengujian Ukuran *offspring*

| Ukuran Offspring | generasi = 50 ; populasi = 30 | | | | | | | | | | rata-rata |
|------------------|-------------------------------|------|------|------|------|------|------|------|------|------|-----------|
| | percobaan ke- | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 1μ | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| | 4243 | 3181 | 2515 | 3545 | 2927 | 3039 | 3191 | 3827 | 3759 | 3239 | 3347 |
| 2μ | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| | 3367 | 352 | 2963 | 3845 | 4049 | 3726 | 3766 | 3193 | 3105 | 3324 | 3486 |
| 3μ | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| | 4 | 3702 | 3021 | 376 | 3608 | 3534 | 3357 | 4114 | 3374 | 396 | 3643 |
| 4μ | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| | 3644 | 4164 | 4267 | 3548 | 3999 | 3691 | 4018 | 4016 | 5181 | 457 | 411 |
| 5μ | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| | 3917 | 4318 | 4139 | 4148 | 4178 | 4816 | 3372 | 4156 | 4064 | 3732 | 4084 |
| 6μ | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| | 3614 | 3739 | 3698 | 4564 | 3852 | 4189 | 3856 | 4824 | 4518 | 5354 | 4221 |
| 7μ | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| | 4611 | 44 | 4192 | 5289 | 4666 | 4183 | 4513 | 4873 | 444 | 3769 | 4494 |
| 8μ | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| | 4318 | 3844 | 4262 | 4505 | 4565 | 4123 | 4429 | 4461 | 396 | 4099 | 4257 |
| 9μ | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| | 4778 | 3739 | 454 | 4089 | 3968 | 4074 | 4597 | 4017 | 5387 | 4558 | 4375 |
| 10μ | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| | 4495 | 3899 | 4616 | 446 | 4989 | 4772 | 3705 | 4328 | 4287 | 4791 | 4434 |

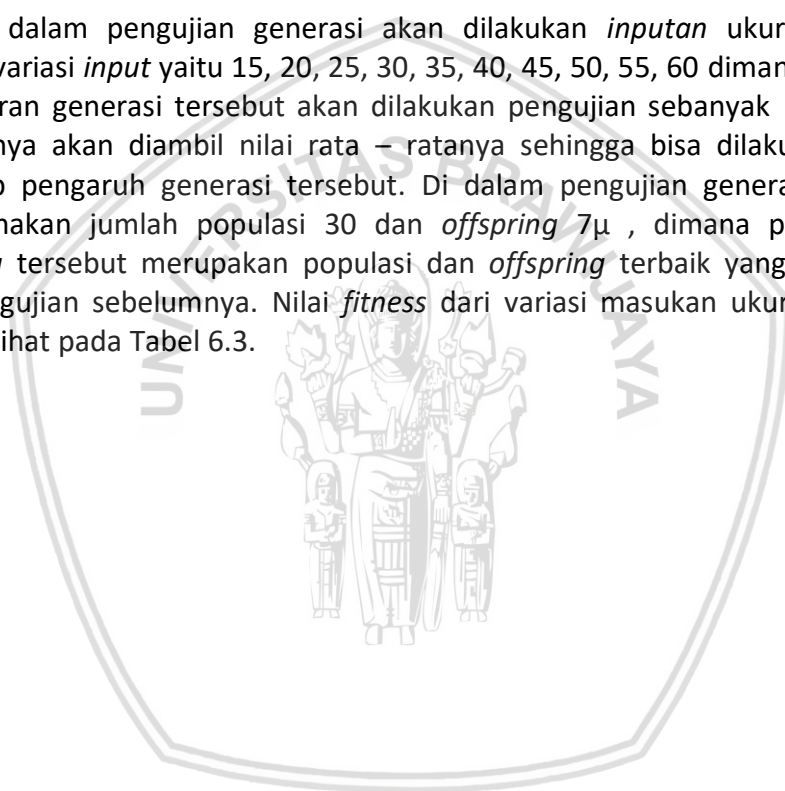
Gambar 0.2 Grafik Pengujian Ukuran *offspring*

Di dalam pengujian *offspring* ini, dilakukan dengan memberikan variasi masukan jumlah *offspring* sebanyak 1 μ , 2 μ , 3 μ , 4 μ , 5 μ , 6 μ , 7 μ , 8 μ , 9 μ , 10 μ dengan generasi sebanyak 50 dan ukuran populasi adalah 30 yang merupakan ukuran populasi terbaik berdasarkan pengujian sebelumnya. Pengujian ini akan

dilakukan masing – masing sebanyak 10 kali yang kemudian diambil nilai rata – rata *fitness* nya dimana nantinya akan terlihat inputan *offspring* terbaik berdasarkan nilai rata – rata *fitness* yang terbesar. Berdasarkan hasil dari Tabel 6.2 dan Gambar 6.2, terlihat bahwa nilai rata – rata *fitness* terbesar terjadi pada saat *offspring* 7 μ yaitu 0,004494. Sedangkan nilai rata – rata *fitness* terkecil yaitu 0,003347 ketika *offspring* 1 μ . Berdasarkan Gambar 6.2 juga terlihat bahwa nilai rata – rata cenderung turun pada saat 5 μ dan 8 μ sehingga hal tersebut menunjukkan bahwa semakin besar ukuran *offspring* tidak memberikan jaminan bahwa nilai *fitness* yang dihasilkan akan meningkat karena inisialisasi kromosom pada sistem dibangkitkan secara acak.

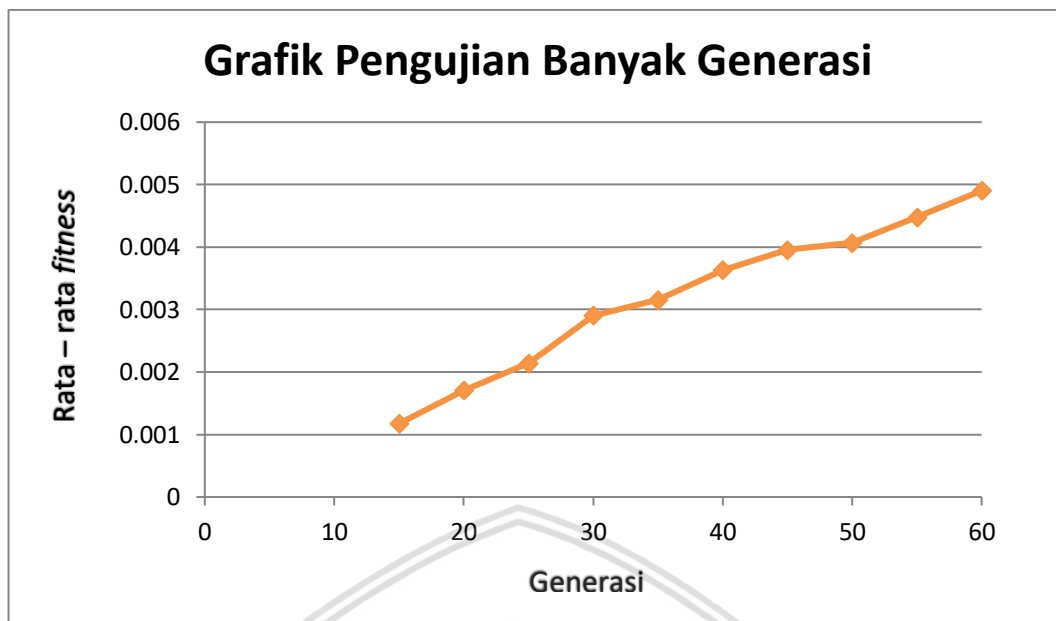
1.2 Hasil dan Analisa Pengujian Banyak Generasi

Di dalam pengujian generasi akan dilakukan *inputan* ukuran generasi dengan variasi *input* yaitu 15, 20, 25, 30, 35, 40, 45, 50, 55, 60 dimana dari tiap – tiap ukuran generasi tersebut akan dilakukan pengujian sebanyak 10 kali yang selanjutnya akan diambil nilai rata – ratanya sehingga bisa dilakukan analisa terhadap pengaruh generasi tersebut. Di dalam pengujian generasi ini, akan menggunakan jumlah populasi 30 dan *offspring* 7 μ , dimana populasi dan *offspring* tersebut merupakan populasi dan *offspring* terbaik yang didapatkan dari pengujian sebelumnya. Nilai *fitness* dari variasi masukan ukuran generasi dapat dilihat pada Tabel 6.3.



Tabel 0.3 Pengujian Banyak Generasi

| Banyak Generas i | Populasi = 30 ; lambda = 7miu | | | | | | | | | | rata - rata |
|------------------------|-------------------------------|---------------------|---------------------|------------------|------------------|------------------|------------------|-------------------|------------------|------------------|-------------------|
| | percobaan ke- | | | | | | | | | | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 15 | 0,00 125 | 0,00 111 8 | 0,00 107 3 | 0,00 125 7 | 0,00 109 7 | 0,00 103 5 | 0,00 118 4 | 0,00 102 7 | 0,00 119 9 | 0,00 149 7 | 0,00 117 4 |
| | 0,00 164 1 | 0,00 169 6 | 0,00 209 3 | 0,00 163 4 | 0,00 180 3 | 0,00 179 8 | 0,00 156 9 | 0,00 164 3 | 0,00 161 8 | 0,00 157 2 | 0,00 170 7 |
| | 0,00 236 1 | 0,00 212 7 | 0,00 207 3 | 0,00 226 2 | 0,00 227 4 | 0,00 209 4 | 0,00 186 4 | 0,00 199 4 | 0,00 232 4 | 0,00 202 4 | 0,00 214 |
| 30 | 0,00 287 8 | 0,00 296 4 | 0,00 305 1 | 0,00 234 8 | 0,00 270 6 | 0,00 298 7 | 0,00 307 4 | 0,00 299 6 | 0,00 302 8 | 0,00 301 | 0,00 290 4 |
| | 0,00 0,00 356 | 0,00 266 3 | 0,00 303 7 | 0,00 296 9 | 0,00 315 1 | 0,00 293 8 | 0,00 315 8 | 0,00 312 3 | 0,00 384 | 0,00 309 7 | 0,00 315 4 |
| | 0,00 381 6 | 0,00 350 5 | 0,00 0,00 349 | 0,00 352 3 | 0,00 361 4 | 0,00 385 8 | 0,00 370 4 | 0,00 332 1 | 0,00 355 6 | 0,00 393 | 0,00 363 2 |
| 45 | 0,00 457 9 | 0,00 383 3 | 0,00 430 5 | 0,00 387 2 | 0,00 414 7 | 0,00 335 6 | 0,00 426 8 | 0,00 366 4 | 0,00 358 6 | 0,00 392 | 0,00 395 3 |
| | 0,00 468 4 | 0,00 0,00 474 | 0,00 373 3 | 0,00 372 7 | 0,00 385 7 | 0,00 405 8 | 0,00 419 7 | 0,00 0,00 4 | 0,00 351 3 | 0,00 419 4 | 0,00 407 |
| | 0,00 511 5 | 0,00 368 9 | 0,00 450 7 | 0,00 391 8 | 0,00 443 8 | 0,00 464 5 | 0,00 496 1 | 0,00 423 1 | 0,00 476 1 | 0,00 449 7 | 0,00 447 6 |
| 60 | 0,00 535 3 | 0,00 450 6 | 0,00 532 2 | 0,00 422 4 | 0,00 462 4 | 0,00 456 9 | 0,00 495 2 | 0,00 455 4 | 0,00 549 5 | 0,00 544 5 | 0,00 490 4 |



Gambar 0.3 Grafik Pengujian Banyak Generasi

Berdasarkan dari Tabel 6.3 dan Gambar 6.3 menunjukkan bahwa hasil dari perbandingan banyak generasi pada nilai *fitness* cenderung meningkat. Sama seperti pada pengujian populasi, berdasarkan pengujian banyak generasi juga terlihat bahwa seiring dengan bertambahnya nilai dari banyaknya generasi, maka nilai *fitness*-nya pun juga akan semakin besar. Hal itu terjadi karena ruang lingkup pencarian akan menjadi lebih luas seiring dengan bertambahnya banyak generasi.

1.3 Analisa Hasil

Berdasarkan hasil serta pembahasan di dalam pengujian yang telah dilakukan maka didapatkan parameter – parameter terbaik seperti ukuran populasi 50, ukuran *offspring* 7μ dan banyak generasi 60. Parameter – parameter terbaik itulah yang akan digunakan untuk dapat menghasilkan suatu rekomendasi dari sistem bagi Atlet UKM Menembak BASIC SC Universitas Brawijaya yang kemudian akan dilakukan perbandingan dengan kebutuhan energi dan zat gizi sebenarnya.

1.3.1 Studi Kasus 1

Rakha Hasan Pramudito berusia 22 tahun, laki – laki dengan berat badan 67 kg, berlatih 3 kali dalam seminggu dan setiap latihan ± 2 jam (120 menit), tingkat aktifitas tergolong aktif ringan. Anggaran yang diinginkan untuk 3 kali makan sebesar Rp 75.000,-

Parameter algoritma *Evolution Strategies* yang digunakan yaitu:

Ukuran Populasi : 50

Ukuran *Offspring* : 7 μ

Banyak Generasi : 60

Dengan menggunakan parameter tersebut maka didapatkan hasil perbandingan kebutuhan energi dan kandungan gizi sebenarnya dengan kebutuhan energi dan kandungan gizi rekomendasi sistem sebagai berikut:

Tabel 0.4 Hasil Pemenuhan Gizi Studi Kasus 1

| | Kebutuhan Atlet | Rekomendasi Sistem | %Gizi Tercukupi | %Selisih Nilai Gizi |
|----------------------------|------------------------|---------------------------|------------------------|----------------------------|
| Karbohidrat | 468 Gr | 431 Gr | 92,1% | 7,9% |
| Lemak | 86,6 Gr | 75,97 Gr | 87,7% | 12,3% |
| Protein | 117 Gr | 120,3 Gr | 102,8% | 2,8% |
| Anggaran Diinginkan | Rp 75.000,- | | | |
| Anggaran Sistem | Rp 73.228,- | | | |

Berdasarkan dari Tabel 6.4 diketahui bahwa sistem mampu memberikan rekomendasi dengan harga minimal dan asupan karbohidrat yang direkomendasikan oleh sistem tercukupi 92,1% , lemak tercukupi 87,7% dan protein tercukupi 102,8%. Dimana karbohidrat serta lemak termasuk kategori cukup dan protein termasuk kategori yang baik dan dapat di toleransi (tidak kurang atau defisit).

1.3.2 Studi Kasus 2

Muhammad Hasbi Ash Shiddieqy berusia 21 tahun, laki – laki dengan berat badan 50 kg, berlatih 2 kali dalam seminggu dan setiap latihan \pm 4 jam (240 menit), tingkat aktifitas tergolong aktif ringan. Anggaran yang diinginkan untuk 3 kali makan sebesar Rp 65.000,-

Parameter algoritma *Evolution Strategies* yang digunakan yaitu

Ukuran Populasi : 50

Ukuran *Offspring* : 7 μ

Banyak Generasi : 60

Dengan menggunakan parameter tersebut maka didapatkan hasil perbandingan kebutuhan energi dan kandungan gizi sebenarnya dengan kebutuhan energi dan kandungan gizi rekomendasi sistem sebagai berikut:

Tabel 0.5 Hasil Pemenuhan Gizi Studi Kasus 2

| | Kebutuhan Atlet | Rekomendasi Sistem | %Gizi Tercukupi | %Selisih Nilai Gizi |
|---------------------|-----------------|--------------------|-----------------|---------------------|
| Karbohidrat | 398 Gr | 360,52 Gr | 90,5% | 9,5% |
| Lemak | 73,8 Gr | 75,3 Gr | 102,03% | 2,03% |
| Protein | 99,6 Gr | 97,12 Gr | 97,5% | 2,5% |
| Anggaran Diinginkan | Rp 65.000,- | | | |
| Anggaran Sistem | Rp 64.451,- | | | |

Berdasarkan dari Tabel 6.5 diketahui bahwa sistem mampu memberikan rekomendasi dengan harga minimal dan asupan karbohidrat yang direkomendasikan oleh sistem tercukupi 90,5% , lemak tercukupi 102,03% dan protein tercukupi 97,5%. Dimana karbohidrat serta protein termasuk kategori cukup dan lemak termasuk kategori yang baik dan dapat di toleransi (tidak kurang atau defisit).

1.3.3 Studi Kasus 3

Eva Yulianti Pratiwi berusia 21 tahun, perempuan dengan berat badan 78 kg, berlatih 2 kali dalam seminggu dan setiap latihan \pm 2 jam (120 menit), tingkat aktifitas tergolong aktif ringan. Anggaran yang diinginkan untuk 3 kali makan sebesar Rp 70.000,-

Parameter algoritma *Evolution Strategies* yang digunakan yaitu

Ukuran Populasi : 50

Ukuran *Offspring* : 7 μ

Banyak Generasi : 60

Dengan menggunakan parameter tersebut maka didapatkan hasil perbandingan kebutuhan energi dan kandungan gizi sebenarnya dengan kebutuhan energi dan kandungan gizi rekomendasi sistem sebagai berikut:

Tabel 0.6 Hasil Pemenuhan Gizi Studi Kasus 3

| | Kebutuhan Atlet | Rekomendasi Sistem | %Gizi Tercukupi | %Selisih Nilai Gizi |
|---------------------|-----------------|--------------------|-----------------|---------------------|
| Karbohidrat | 442,5 Gr | 377,43 Gr | 85,3% | 14,7% |
| Lemak | 81,9 Gr | 93,89 Gr | 114,6% | 14,6% |
| Protein | 110,6 Gr | 121,8 Gr | 110,1% | 10,1% |
| Anggaran Diinginkan | Rp 70.000,- | | | |
| Anggaran Sistem | Rp 69.691,- | | | |

Berdasarkan dari Tabel 6.6 diketahui bahwa sistem mampu memberikan rekomendasi dengan harga minimal dan asupan karbohidrat yang direkomendasikan oleh sistem tercukupi 85,3% , lemak tercukupi 114,6% dan protein tercukupi 110,1%. Dimana karbohidrat termasuk kategori cukup, sedangkan protein dan lemak termasuk kategori yang baik dan dapat di toleransi (tidak kurang atau defisit).

1.3.4 Studi Kasus 4

Yulia Anggita Girsang berusia 20 tahun, perempuan dengan berat badan 49 kg, berlatih 5 kali dalam seminggu dan setiap latihan ± 2 jam (120 menit), tingkat aktifitas tergolong aktif ringan. Anggaran yang diinginkan untuk 3 kali makan sebesar Rp 60.000,-

Parameter algoritma *Evolution Strategies* yang digunakan yaitu

Ukuran Populasi : 50

Ukuran *Offspring* : 7μ

Banyak Generasi : 60

Dengan menggunakan parameter tersebut maka didapatkan hasil perbandingan kebutuhan energi dan kandungan gizi sebenarnya dengan kebutuhan energi dan kandungan gizi rekomendasi sistem sebagai berikut:

Tabel 0.7 Hasil Pemenuhan Gizi Studi Kasus 4

| | Kebutuhan Atlet | Rekomendasi Sistem | %Gizi Tercukupi | %Selisih Nilai Gizi |
|----------------------------|-----------------|--------------------|-----------------|---------------------|
| Karbohidrat | 352 Gr | 325,6 Gr | 92,4% | 7,6% |
| Lemak | 65 Gr | 59,16 Gr | 90,6% | 9,4% |
| Protein | 88 Gr | 97,91 Gr | 111,1% | 11,1% |
| Anggaran Diinginkan | Rp 60.000,- | | | |
| Anggaran Sistem | Rp 58.905,- | | | |

Berdasarkan dari Tabel 6.7 diketahui bahwa sistem mampu memberikan rekomendasi dengan harga minimal dan asupan karbohidrat yang direkomendasikan oleh sistem tercukupi 92,4% , lemak tercukupi 90,6% dan protein tercukupi 111,1%. Dimana karbohidrat serta lemak termasuk kategori cukup dan protein termasuk kategori yang baik dan dapat di toleransi (tidak kurang atau defisit).

Tabel 0.8 Hasil Persentase Kecukupan Gizi Berdasarkan Studi Kasus

| | Studi Kasus 1 | Studi Kasus 2 | Studi Kasus 3 | Studi Kasus 4 | Rata – Rata Tercukupi |
|--------------------|---------------|---------------|---------------|---------------|-----------------------|
| Karbohidrat | 92,1% | 90,5% | 85,3% | 92,4% | 90,1% |
| Lemak | 87,7% | 102,03% | 114,6% | 90,6% | 98,7% |
| Protein | 102,8% | 97,5% | 110,1% | 111,1% | 105,4% |

Tabel 0.9 Hasil Perbandingan Harga Berdasarkan Studi Kasus

| | Studi Kasus 1 | Studi Kasus 2 | Studi Kasus 3 | Studi Kasus 4 |
|----------------------------|---------------|---------------|---------------|---------------|
| Anggaran Diinginkan | Rp 75.000,- | Rp 65.000,- | Rp 70.000,- | Rp 60.000,- |
| Anggaran Sistem | Rp 73.228,- | Rp 64.451,- | Rp 69.691,- | Rp 58.905,- |

Berdasarkan dari pengujian sistem terhadap 4 studi kasus, terlihat bahwa sistem mampu memberikan rekomendasi berat yang baik pada setiap pengujian sistem terhadap masing – masing studi kasus. Sehingga diketahui bahwa rata – rata hasil rekomendasi sistem masih termasuk ke dalam indikator yang tercukupi dan dapat di toleransi (tidak kurang atau defisit) menurut para ahli gizi dalam melakukan penilaian konsumsi pangan dengan rata – rata karbohidrat tercukupi 90,1%, lemak 98,7% dan protein 105,4%, dimana karbohidrat serta lemak termasuk kategori cukup dan protein termasuk kategori yang baik. Selain itu harga berdasarkan rekomendasi makanan oleh sistem lebih rendah dibandingkan dengan anggaran yang diinginkan oleh *user*.



PENUTUP

Bab ini berisi mengenai kesimpulan dari perancangan, implementasi dan pengujian, serta saran untuk digunakan pada penulisan selanjutnya.

1.1 Kesimpulan

Berdasarkan dari penelitian yang telah dilakukan maka dapat disimpulkan bahwa di dalam pengimplementasian optimasi komposisi bahan makanan atlet menembak menggunakan algoritma *Evolution Strategies* yang pertama yaitu dengan menentukan panjang serta pengkodean kromosom yang akan digunakan dalam penyelesaian masalah optimasi ini. Pada penelitian ini, panjang kromosom ditentukan berdasarkan banyaknya bahan makanan yang akan di-*input*-kan oleh *user*, dimana kromosom tersebut merepresentasikan berat bahan makanan yang didapatkan secara random oleh sistem. Kemudian selanjutnya adalah proses mutasi dengan tujuan menghasilkan *offspring*. Setelah proses mutasi tersebut maka akan didapatkan kromosom dari *offspring* dan juga *parent* yang kemudian dilakukan perhitungan nilai *fitness* dari masing – masing individu *offspring* dan *parent*. Apabila telah didapatkan nilai *fitness* tersebut maka akan dilakukan proses seleksi dengan menggunakan *elitism* yang mana proses seleksi ini melibatkan hasil *fitness offspring* dan *parent* kemudian diurutkan berdasarkan individu dengan nilai *fitness* terbesar hingga terkecil yang mana akan dipilih untuk masuk ke generasi selanjutnya dan proses akan berhenti jika telah mencapai banyak generasi yang dimasukan oleh *user* dimana solusi terbaik didapatkan berdasarkan dari nilai *fitness* yang terbesar.

Kedua, dalam menyelesaikan permasalahan optimasi komposisi bahan makanan atlet menembak, parameter – parameter yang terdapat di dalam algoritma *Evolution Strategies* cukup mempengaruhi hasil yang akan dihasilkan oleh sistem. Berdasarkan dari pengujian maka didapatkan hasil ukuran populasi terbaik yaitu 30, *offspring* terbaik yaitu pada saat 7 μ dan generasi terbaik sebanyak 60. Dari hasil pengujian itu pula terlihat bahwa penentuan parameter sangat berpengaruh terhadap nilai *fitness* yang dihasilkan sebab penambahan ukuran atau jumlah parameter – parameter tersebut sangat memungkinkan untuk memperluas area pelacakan solusi namun waktu komputasi juga akan semakin meningkat, selain itu nilai *fitness* yang dihasilkan juga tidak menjamin kenaikan rata – rata nilai *fitness* yang signifikan, hal ini disebabkan oleh konsep *random* pada algoritma *Evolution Strategies* yang mana pembangkitan populasi awal dan *strategy parameter* dilakukan secara *random*.

Ketiga, berdasarkan dari hasil pengujian sistem terhadap 4 studi kasus, diketahui bahwa rata – rata hasil rekomendasi sistem masih termasuk ke dalam indikator yang tercukupi dan dapat di toleransi (tidak kurang atau defisit) menurut para ahli gizi dalam melakukan penilaian konsumsi pangan dengan rata – rata energi tercukupi 90%, karbohidrat 90,1%, lemak 98,7% dan protein 105,4%. Selain itu harga berdasarkan rekomendasi makanan oleh sistem lebih rendah dibandingkan dengan anggaran yang diinginkan oleh *user*.

1.2 Saran

Saran yang akan diberikan guna melakukan pengembangan sistem lebih lanjut yaitu:

1. Mencoba menggunakan rumus perhitungan gizi yang lain agar hasil rekomendasi sistem dapat diperuntukan bagi atlet dengan fase periodisasi lainnya misal untuk atlet yang memasuki fase persiapan khusus atau bisa juga yang sedang memasuki fase *pra* pertandingan pada hari ke – 3, 2, 1 sebelum pertandingan dengan metode *carbohydrat loading*.
2. Menambahkan daftar makanan yang akan digunakan agar daftar makanan yang dipilih *user* bisa lebih bervariasi.
3. Menggunakan siklus lain dari algoritma *Evolution Strategies* atau melakukan hibridasi algoritma *Evolution Strategies* dengan algoritma yang lainnya guna memperbaiki solusi.
4. Menggunakan rumus perhitungan nilai *fitness* yang lebih baik lagi serta dengan menambahkan nilai – nilai parameter pada pengujian sistem yang lebih beragam agar lebih luas ruang lingkup pencariannya.



DAFTAR PUSTAKA

- Anonim., 2011. *Saatnya Ubah Pola Makan Kita* [online]. Tersedia di < <https://obesolution.wordpress.com/2011/03/> > [Diakses 26 Oktober 2017]
- Apfel Glenn,C., 2011. *"MERAH EMAS Di Olympic Youth Games 2014 dan 2018: Pengenalan Dasar Cabang Olahraga Menembak Seri Junior & Youth"*. Jakarta
- Damajanti,M, dkk., 2014. *"Pedoman Gizi Olahraga Prestasi"*. Kementerian Kesehatan RI. Jakarta
- Herdanis,C , Chollisodin,I & Fauzi,MA., 2016. *"Optimasi Gizi Pada Anak Panti Asuhan Menggunakan Evolution Strategies"*. Skripsi. Universitas Brawijaya. Malang
- Lala,M., 2011. *Penilaian Konsumsi Pangan* [online]. Tersedia di < <https://gizimu.wordpress.com/2011/10/22/penilaian-konsumsi-pangan-2/> > [Diakses 03 Mei 2018]
- Mahmudy,Wf., 2015. *"Dasar – Dasar Algoritma Evolusi"*. Universitas Brawijaya. Malang
- Mentari,M , Sari,EKR & Mutrofin,S., 2014. *"Klasifikasi Menggunakan Kombinasi Multilayer Perception dan Alignment Particle Swarm Optimization"*. Bangkalan, Seminar Nasional Teknologi Informasi & Komputasi 2014.
- Peraturan Menteri Kesehatan Republik Indonesia (PMK RI) Nomor 41 Tahun 2014 Tentang Pedoman Gizi Seimbang. Jakarta: Menteri Kesehatan Republik Indonesia.
- Purwa,IDMAP , Budjana,IGB & Sugiantara,IP., 2015. *"Pusat Pelatihan Dan Sarana Olahraga Menembak Di Denpasar"*. Jurnal Teknik Arsitektur. Universitas Udayana. Bali
- Rachmat,ZY , Ratnawati,DE & Arwan,A., 2016. *"Optimasi Komposisi Makanan Untuk Atlet Endurance Menggunakan Metode Particle Swarm Optimization"*. Jurnal Teknik Informatika". Universitas Brawijaya. Malang
- Rahmadhani,N , Ratnawati,DE & Data,M., 2016. *"Optimasi Komposisi Makanan Untuk Atlet Olahraga Endurance Menggunakan Algoritma Genetika"*. Skripsi. Universitas Brawijaya. Malang
- SISKAPERBAPO., 2017. *Sistem Informasi Ketersediaan dan Perkembangan Harga Bahan Pokok di Jawa Timur* [online]. Jawa Timur: Dinas Perindustrian dan Perdagangan (Disperindag). Tersedia di < <http://siskaperbapo.com/harga/tabel> > [Diakses 19 Oktober 2017]
- Sudarko,RA., 2009. *"Peningkatan Kualitas Prosedur dan Evaluasi Olahraga Unggulan Propinsi Kalimantan Timur"*. Jurnal Olahraga Prestasi Volume 5, No. 1 Januari 2009. Universitas Negeri Yogyakarta. Yogyakarta

- Suryandriyo,B., 2013. *Pengertian atau Arti Makna dan Definisi Olahraga Secara Umum* [online]. Tersedia di <
<http://www.ikerenki.com/2013/12/pengertian-arti-makna-definisi-olahraga-menurut-ahli-pakar.html> > [Diakses 2 Maret 2017]
- Yansari,M , Ratnawati,DE & Marji., 2016. *“Optimasi Biaya dan Asupan Gizi Pasien Diet Khusus dengan Menggunakan Algoritma Evolution Strategies”*. Universitas Brawijaya. Malang

